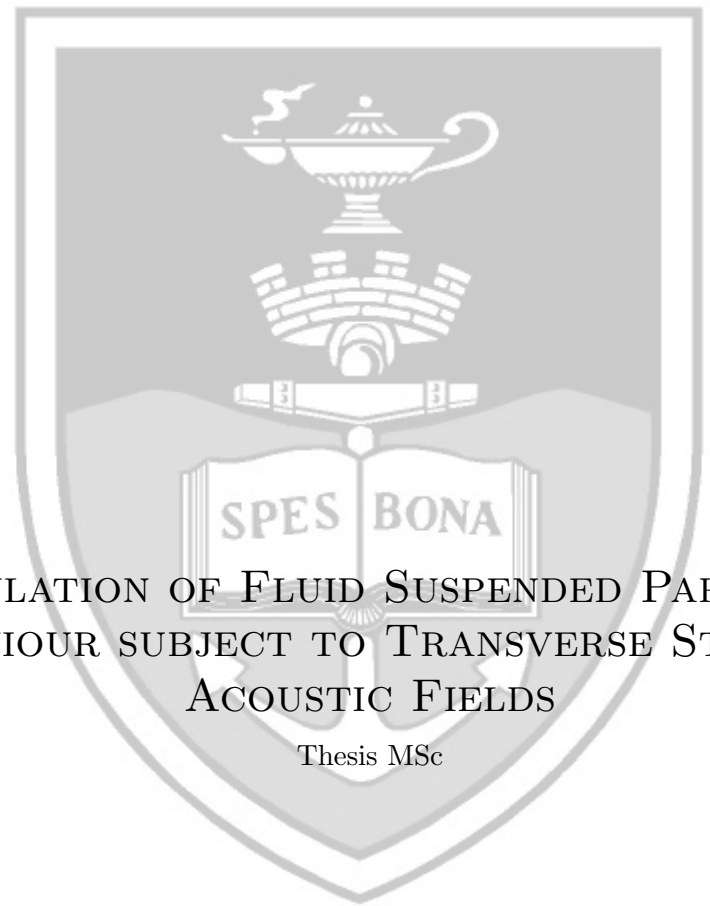
Mihajlo Dabic



Simulation of Fluid Suspended Particle Behaviour subject to Transverse Standing Acoustic Fields

Thesis MSc

University of Cape Town

December 11, 2012

University of Cape Town

Center for Research in Computational and Applied Mechanics

Simulation of Fluid Suspended Particle Behaviour subject to Transverse Standing Acoustic Fields

---

THESIS

Presented to the Faculty of Chemical Engineering in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

Supervisors:

Prof. D.A. Deglon .....

Prof. C.J. Meyer .....

Candidate:

Mr M. Dabic .....

December 11, 2012

I know the meaning of plagiarism and declare that all the work in the document, save for that which is properly acknowledged, is my own

Dedication

To honesty, decency, fairness and kindness

Abstract

The computational study addressed the effectiveness with which a standing wave acoustic field could be used to deflect quartz particles carried in water at 20 °C through a simple parallelepiped control volume representative of a vertically orientated duct geometry dimensioned $50 \times 50 \times 70$ cm^3, with square base. An acoustically driven planar standing wave field produces quasi-static oscillatory pressure gradients within resonant cavities, which are responsible for acoustic forces which act on particles located within the acoustic field. These forces drive particles to nodal (no fluctuation) or anti-nodal (continuous fluctuation) planes of pressure. Standing wave fields are generally produced by a transducer driving into a fluid through an adhesively bonded matching layer. The wave is reflected at the opposite boundary layer terminating in an air backing. The chamber is dimensioned so as to produce constructive wave interference between the two waves travelling in opposite directions. The acoustic force has been used in small scale filtration systems to deflect particles and on larger scales as a pre filtration agglomerator clumping very small particles which are otherwise poorly filtered in isolation by conventional methods. The study was two fold, in that a major component of the study comprised developing the architecture of the computational model, the other part comprising qualitative model validation through parameter variation.

The study involved coupling between Computational Fluid Dynamics (CFD) Software (OpenFOAM) and Discrete Element Modelling (DEM) Software (LIGGGHTS), through a coupling code (CFDEM) built as an extension to OpenFOAM and tailored for LIGGGHTS. The acoustic field was assumed ideal i.e. in a lossless medium with perfect reflection at the opposite wall. Particle-particle and particle-wall collisions were circumvented by using larger time increments, inadequate to resolve collisions, and inserting particles in the bulk of the flow away from any wall boundary. Twenty particles with uniform radial size distribution in the range 5-30 micron were seeded in the flow field about 10 cm from the bottom inlet, and carried in the z direction at various flow speeds, 0.1 ms^{-1}, 0.5 ms^{-1} and 1 ms^{-1}, whilst being subject to acoustic forces in the x direction, to investigate deflection response and transducer lengths required to achieve adequate lateral deflection. The model accounted for drag, buoyancy, gravity and primary acoustic forces. Flow velocities distinguished by those maximum velocities recorded at duct centrelines were obtained by adjusting pressure gradients across the domain. The fluid continuum was modelled through Reynolds Averaged Navier Stokes (RANS) equations, supplemented by an eddy viscosity $k - \epsilon$ two equation turbulence model. The flow profile was validated against the analytic Darcy-Weisbach pressure to mean velocity relation. Two acoustic driving frequencies, 14794 Hz and 26629 Hz, were investigated for each flow rate to determine the effect frequency had on acoustic force magnitude, nodal distribution and particle residence time. Acoustic deflection efficiency was measured as that time or particle vertical travel length required, coinciding with a lateral deflection to within 1.5 mm of an adjacent nodal plane. From a computational point of interest acoustic force dependencies and trends were qualitatively evaluated for consistency with theoretic equations and published literature.

The major limitations of the study arose from the ideal standing wave assumption and the inability of the $k - \epsilon$ model to resolve cross currents and secondary flows. In reality the acoustic field would be

highly non-linear due to absorptive effects and particles would interact continuously with turbulent eddies and secondary currents perpendicular to the primary flow direction, which may or may not affect the function of the acoustic force. The acoustic model is however applicable without modification to regular shaped microfluidic channels in which flow is typically laminar, devoid of eddying currents, and subject to minimal acoustic attenuation.

The study concluded that the acoustic force could be used to deflect particles in the size range investigated, however under the investigated parameters no particularly practical separation distance was achieved. General trends noted in literature and consistent with the governing equation form of the acoustic force have been observed i.e. larger particles are more effectively deflected, dense incompressible particles are deflected toward nodal planes, higher driving frequencies produce larger forces and more closely spaced nodal planes. Driving amplitude was not varied hence comments were reserved in this regard. Varied particle density in the limited range was found to have little effect on the strength of acoustic force however its contribution to the acoustic equation was in terms of governing whether particles were forced to nodal or anti nodal planes. Low frequency high amplitude systems would be more suited to producing larger particle deflections whereas high frequency systems producing closely spaced nodal planes would only be useful in agglomerating applications as pre filtration components. Lower flow rates allowed the required residence times to be more readily realized. Residence time was found to be an important measure of deflection i.e. that time a particle must be subject to an acoustic field to align with its nearest pressure node to within a given tolerance distance.

ACKNOWLEDGEMENTS

I hereby acknowledge the contributions toward the completion of the following study by individuals, institutions and organizations: Minerals To Metals (South Africa) and the Department of Chemical Engineering at the University of Cape Town for providing funding; Prof. D.A. Deglon and Prof. C.J. Meyer for acting as supervisory authorities and producing the foundation and bounds of the study; Evan Smuts (CERECAM) for providing certain code segments used in simulation work as well as in his assistance in resolving DEM simulation issues and Jonathan Bergh (CERECAM) in providing input with respect to turbulence modelling. The completion of the study would not have been possible without the aforementioned contributors.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF CODE FRAGMENTS

NOMENCLATURE

$()^T$	transpose operator
α	attenuation coefficient
α_f	fluid to solid void or volume fraction
$\bar{u}$	magnitude of average velocity in the primary flow direction (m s^{-1})
$\overline{\mathbf{p_{in}^2}}$	mean square instantaneous pressure fluctuation at a point where the particle is located (kg^2 m^{-2} s^{-4})
$\overline{\mathbf{u_{in}^2}}$	mean square instantaneous velocity fluctuation at a point where the particle is located (m^2 s^{-2})
$\overline{\mathbf{U}}$	ensemble average velocity component (m s^{-1})
β	contact force coefficient
β_f	fluid medium compressibility (kg^{-1} m s^2)
β_p	particle material compressibility (kg^{-1} m s^2)
β_T	fluid medium Bulk Modulus (kg m^{-1} s^{-2})
β_{p1}	compressibility of particle material 1 (kg^{-1} m s^2)
β_{p2}	compressibility of particle material 2 (kg^{-1} m s^2)
$\boldsymbol{\omega}$	fluid vorticity (s^{-1})
$\cdot$	contraction operator
χ	void fraction coefficient
$\delta\mathbf{x}_{p,n}$	normal spatial contact overlap between contacting particles (m)
ΔP	pressure drop across pipe or duct segment (kg m^{-1} s^{-2})
Δt	DEM time step (s)
δt	time increment (s)
δx	mesh cell dimension in the primary flow direction (m)
ℓ	turbulent length scale (m)
ϵ	energy dissipation rate (m^2 s^{-3})
ϵ_{ac}	acoustic energy density (kg m^{-1} s^{-2})
η	coefficient of dynamic shear viscosity (kg m^{-1} s^{-1})

γ	ratio of specific heats
$\int$	integral operator
κ	Von Karman constant 0.41
λ	wave length (m)
μ	dynamic fluid viscosity (kg m^{-1} s^{-1})
μ_c	coefficient of static friction
μ_e	effective fluid dynamic viscosity (kg m^{-1} s^{-1})
∇	gradient or divergence operator
ν	kinematic fluid viscosity (m^2 s^{-1})
ν_t	kinematic turbulent fluid viscosity (m^2 s^{-1})
$\nu_{1,2}$	Poissons Ratio of sphere material 1 or 2 in contact
ν_e	effective kinematic fluid viscosity (m^2 s^{-1})
ω	angular driving frequency (s^{-1})
ω^*	lift force coefficient
$\Omega^*_{p,eq}$	particle spin rate coefficient
$\otimes$	diadic operator
∂	partial derivative operator
ϕ	surface velocity flux
ρ	fluid density (kg m^{-3})
ρ_E	cohesion energy density (kg m^{-1} s^{-2})
ρ_f	fluid density (kg m^{-3})
ρ_p	particle material density (kg m^{-3})
$\sum$	summation operator
τ	time integral (s)
τ_d	diffusive time scale (s)
τ_w	wall shear stress (kg m^{-1} s^{-2})
$\times$	cross product operator
$\mathbf{A}$	diagonal matrix of the velocity related coefficient array

$\mathbf{a}$ particle acceleration vector (m s^{-2})

$\mathbf{A}_n$ normal contact area between contacting particles (m^2)

$\mathbf{C}$ compliance matrix

dev deviatoric operator

$\mathbf{D}$ rate of deformation tensor (s^{-1})

$\mathbf{F_{ac}}$ vector acoustic force ((kg m^{-1} s^{-2}))

$\mathbf{F}$ additional body forces (m s^{-2})

$\mathbf{F}_1$ particle buoyancy force vector (kg m s^{-2})

$\mathbf{F}_2$ particle drag force vector (kg m s^{-2})

$\mathbf{F}_B$ Basset force on particle (kg m s^{-2})

$\mathbf{F}_C$ contact force on particle (kg m s^{-2})

$\mathbf{F}_D$ drag force on particle (kg m s^{-2})

$\mathbf{F}_f$ fluid force on suspended particle corresponding to subscript f = 1,2,..,N (kg m s^{-2})

$\mathbf{F}_k$ forces acting on particles, k $\in$ 1,2,.. (kg m s^{-2})

$\mathbf{F}_M$ Magnus lift force on particle (kg m s^{-2})

$\mathbf{F}_n$ normal contact force vector (kg m s^{-2})

$\mathbf{F}_P$ pressure gradient force on particle (kg m s^{-2})

$\mathbf{F}_S$ Saffman lift force on particle (kg m s^{-2})

$\mathbf{F}_t$ tangential contact force vector (kg m s^{-2})

$\mathbf{F}_V$ virtual mass force on particle (kg m s^{-2})

$\mathbf{F}_{...}$ additional relevant forces on particle unaccounted for in the formulation (kg m s^{-2})

$\mathbf{F}_{GB}$ gravity buoyancy force on particle (kg m s^{-2})

$\mathbf{F}_i$ implicit forces on a given particle (kg m s^{-2})

$\mathbf{F}_{SM}$ combined Saffman Magnus lift force on particle (kg m s^{-2})

$\mathbf{g}$ gravity vector (m s^{-2})

$\mathbf{H}'$ off-diagonal matrix of the velocity related coefficient array

$\mathbf{R}$ Reynolds Stress Tensor (m^2 s^{-2})

$\mathbf{r}$ source terms

$\mathbf{R}_{sl}$ force density exerted by particles on fluid (m s^{-2})

$\mathbf{S}_f$ face area vector

$\mathbf{U}'$ fluctuating velocity component (m s^{-1})

$\mathbf{U_f}$ fluid velocity vector at particle position (m s^{-1})

$\mathbf{U_G}$ vector Gorkov potential (kg m^2 s^2)

$\mathbf{U_p}$ particle velocity vector (m s^{-1})

$\mathbf{U}$ velocity component vector (m s^{-1})

$\mathbf{U}_r$ relative fluid particle velocity (m s^{-1})

$\mathbf{u}_{p,n}$ normal relative velocity of contacting particles (m s^{-1})

$\mathbf{u}_{p,t}$ relative tangential velocity of contacting particles (m s^{-1})

$\mathbf{v}$ particle velocity vector (m s^{-1})

$\mathbf{x}$ particle position vector (m)

a radius of particle (m)

A_c cross sectional area (m^2)

A_+ displacement amplitude of +x travelling wave (m)

A_- displacement amplitude of -x travelling wave (m)

$A_\pm$ displacement amplitude of both +x and -x travelling waves (m)

B^+ logarithmic layer constant 5.0

c speed of sound in medium (m s^{-1})

c_1 Kernel coefficient 2.5

c_2 Kernel coefficient 0.2

C_D drag coefficient

c_f speed of sound in the fluid (m s^{-1})

C_L combined fluid vorticity particle rotation lift coefficient

c_n normal viscoelastic damping coefficient (kg s^{-1})

c_p speed of sound in the particle material (m s^{-1})

c_t tangential viscoelastic damping coefficient (kg s^{-1})

$C^*_{L,\Omega}$ lift force coefficient

Co	Courant Number
d	total derivative operator
D_H	hydraulic diameter (m)
D_H	hydraulic diameter (m)
d_p	particle diameter (m)
E	particle material Elastic Modulus (kg m^{-1} s^{-2})
e	coefficient of restitution
exp	exponential operator
f	resonant driving frequency (Hz)
f_D	Darcy friction factor
f_H	Basset force coefficient
$F_{ac,2}$	secondary acoustic force (kg m s^{-2})
G^*	effective shear modulus of contact (kg m^{-1} s^{-2})
h_b	sample bounce height (m)
h_d	sample drop height (m)
I	acoustic field intensity (kg s^{-3})
I_0	unattenuated acoustic field intensity (kg s^{-3})
I_t	turbulent intensity
I_x	acoustic field intensity as a function of x coordinate (kg s^{-3})
J^*	lift force coefficient
K	Basset force integration kernel
k	turbulent kinetic energy (m^2 s^{-2})
k_n	normal contact stiffness (kg s^{-2})
k_t	shear contact stiffness (kg s^{-2})
k_{ac}	acoustic wavenumber (m^{-1})
L	length of pipe or duct segment (m)
L_{12}	distance between particle centres (m)
ln	natural logarithm

log_{10}	base 10 logarithmic operator
m^*	effective mass of contact (kg)
m_p	particle mass (kg)
$m_{1,2}$	mass of sphere 1 or 2 of contact pair (kg)
p	pressure at spatial coordinate x and time t (kg m^{-1} s^{-2})
P_1	pressure at position x_1 (kg m^{-1} s^{-2})
P_2	pressure at position x_2 (kg m^{-1} s^{-2})
P_c	cross sectional perimeter (m)
P_k	production rate of turbulent kinetic energy (m^2 s^{-3})
r	centre to centre distance between spheres 1 and 2 (m)
R^*	effective radius of contact (m)
$R_{1,2}$	radius of sphere 1 or 2 of contact pair (m)
Re	Reynolds Number
Re_p	particle Reynold number
Re_ω	particle spin Reynolds number
Re_{D_H}	Reynolds Number based on hydraulic diameter
T	period of oscillation (s)
t	time (s)
t	time integral until loss of particle contact (s)
T_H	Hertz collision contact time (s)
T_R	Rayleigh surface wave propagation time (s)
$t_{c,0}$	time of particle contact (s)
tr	trace operator
u	velocity at spatial coordinate x and time t (m s^{-1})
u'	magnitude of isotropic velocity fluctuation (m s^{-1})
u^+	u plus dimensionless velocity
u_τ	friction or shear velocity (m s^{-1})
V_0	volume of a fluid element (m^3)

V_m	mean velocity (m s^{-1})
v_r	relative velocity of colliding bodies (m s^{-1})
v_{A1}	velocity of material sample A before contact (m s^{-1})
v_{A2}	velocity of material sample A post contact (m s^{-1})
v_{B1}	velocity of material sample B before contact (m s^{-1})
v_{B2}	velocity of material sample B post contact (m s^{-1})
V_{max}	maximum velocity (m s^{-1})
V_{p1}	volume of particle 1 (m^3)
V_{p2}	volume of particle 2 (m^3)
x	x coordinate (m)
x_{i0}	x coordinate of pressure field zeros (m)
Y^*	effective modulus of contact (kg m^{-1} s^{-2})
y^*	y star comparable to y plus in the logarithmic layer
y^+	y plus dimensionless distance to the nearest wall
$Y_{1,2}$	Youngs Modulus of sphere material 1 or 2 in contact (kg m^{-1} s^{-2})
y_{c1}	normal distance from wall adjacent cell centre to nearest wall (m)
δ_n	scalar normal overlap between two particles (m)
ΔE	total energy of a fluid volume element (kg m^2 s^{-2})
ΔE_k	kinetic energy of a volume element (kg m^2 s^{-2})
ΔE_p	potential energy a fluid volume element (kg m^2 s^{-2})
Φ	acoustic contrast factor
σ_k	$k-\epsilon$ constant 1.0
σ_ϵ	$k-\epsilon$ constant 1.3
$C_{1\epsilon}$	$k-\epsilon$ constant 1.44
$C_{2\epsilon}$	$k-\epsilon$ constant 1.92
C_μ	$k-\epsilon$ constant 0.09
F_{ac}	primary acoustic force (kg m s^{-2})
l_x	chamber dimension x (m)

l_y chamber dimension y (m)

l_z chamber dimension z (m)

n_x numbers of natural oscillations in x dimension

n_y numbers of natural oscillations in y dimension

n_z numbers of natural oscillations in z dimension

CHAPTER 1

INTRODUCTION

1.1 Background

Acoustic phenomena have been known to exist for many years, since the development of musical instruments. Forces on solid particles carried in a fluid or gas phase in the presence of an acoustic field have been noted as far back as 1866 by Kundt [3] who observed dust particle deflection in resonant tubes. The field of acoustics was significantly forwarded by Lord Rayleigh's 1877 publication *The Theory of Sound* [51]. The first theoretical model to describe the acoustic force however, was made by King [19] and published in 1934. King's theory however did not account for sphere compressibility. The formulation at the time was able to describe analytically the acoustic force acting on a sphere in a one dimensional planar standing acoustic wave field. Embleton in his 1954 paper described an extension of the theory to spherical and cylindrical fields. This theory was built upon by Yosioko and Kawasima, their paper published in 1955, in which sphere compressibility was incorporated into the original formulation [3]. Gorkov's work in his 1961 paper documented a novel approach to the problem by formulating a potential field which acted on particles forcing them to regions of high or low potential and thereby allowed arbitrary and complex field-particle interaction to be mathematically expressed [3]. Kuznetsov's 'Equation of Non-Linear Acoustics' 1971 paved the way for numerical computation of non-linear three dimensional acoustic fields [5].

Early work on the subject had been limited to solid particle studies and to some extent included liquid droplets but had largely omitted studies involving bubble response in acoustic fields. By the early 1990's particles sized 0.1 - 100 microns had been successfully trapped in experiments against carrier fluid drag using the acoustic force [51]. Doninikov improved on Bjerknes theory describing in more detail mutual interaction forces between oscillating bubbles in resonant fields [51]. Bessel type equations have been used in more recent work describing non-linear acoustic field interaction with particles. More computational modelling studies have been undertaken in recent years geared at many different systems from macro to micro involving all types of particles. In addition to simulating particle trajectories many layered resonator models and models describing non-linear fluid pressure fields have been developed. These have proven useful in paving the way for more efficient chamber design by isolating and quantifying parametric design contributions and relating them to pressure field effects. Not only have computational methods been used to produce more realistic pressure fields but also to optimize actuator design [16]. Macro scale systems are subject to acoustic non-linearities

and hence computational models describing pressure fields within industrial sized cavities have found much use [5]. Experimental work has not become obsolete and continues to be used extensively in validating computational results. Numerous patents have been issued on devices implementing the acoustic force to achieve filtration. More recently computational models have been designed to account for non-linear acoustic wave interaction with nonhomogeneously distributed resonating bubbles [38].

1.2 Fundamental Principles

The acoustic force which acts on suspended particles in acoustic fields is strongest when the encapsulating chamber is in resonance. These one dimensional planar acoustic standing pressure fields are generated within gaseous or fluid media using a transducer, matching layer and reflector arrangement as illustrated in figure *1.1*. The matching layer drives into the adjacent fluid causing fluid molecules to compress and rarefact, setting up a pressure field in the fluid layer. The layered resonator assembly may be mathematically modelled as a one dimensional Resistor Inductor Capacitor (RLC) series circuit in parallel with a static capacitance [48]. This approach had been undertaken by many researchers in the past and continues to be used today to give insight into effects which various parameters have on resulting pressure fields. Structural members must be rigid in order to maintain the integrity of the pressure field. The shape of the acoustic chamber is also important i.e. regular shapes are more conducive to facilitating strong standing wave fields [16].

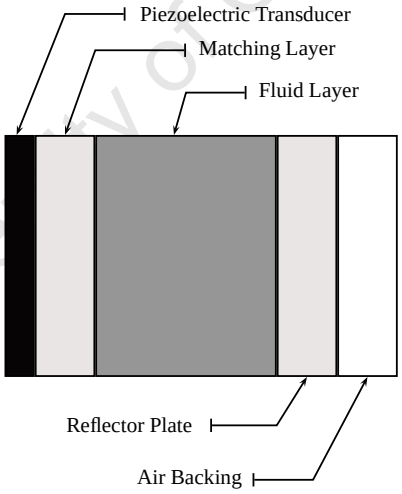


Fig. (1.1) *Layered resonator adapted from [14][48]*

The transducer is generally made from a piezoelectric material which strains under an applied voltage. A common piezoelectric material is Lead Zirconate Titanate (PZT). The PZT material is bonded using adhesive to a matching layer which separates it from the fluid medium. A reflector layer which often terminates in an air backing is used to reflect a pressure wave back through the fluid medium thereby creating a standing pressure field. Typical frequencies used in small scale units range between 0.5 and 15 MHz in liquids, however sonic frequencies i.e. below 20 kHz, have also been used in industrial applications. Generally frequencies around 20 kHz, marking the lower limit of the ultrasonic range, are used in large scale chambers [16]. Large scale applications make use of a different transducer design,

one typically in immediate contact with the fluid and illustrated in figure *1.2*. The figure represents a cross sectional view, the radiating plate can be round or rectangular. The vibrating arrangement comprising sandwich and amplifier are typically cylindrical. The principles of operation in terms of generating the standing wave field nevertheless remain the same. Quality standing wave fields can be produced when the system has good impedance matching with the carrier medium, when achievable transducer displacement amplitudes are relatively large and when the acoustic beam is uniformly distributed and focused.

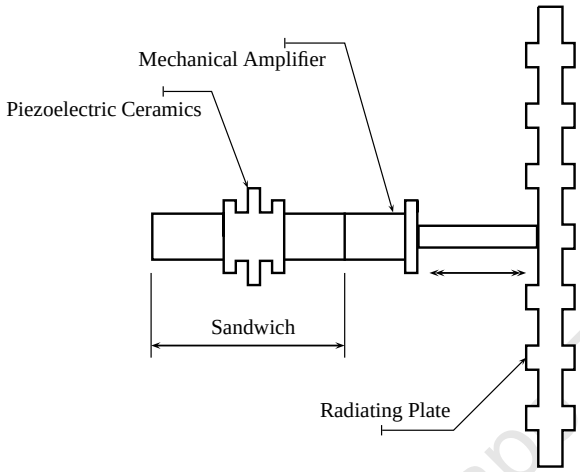


Fig. (1.2) *Acoustic transducer design used in large scale agglomeration application, adapted from [16]*

The acoustic force is superimposed on the standing wave field, as depicted in figure *1.3*. Blue lines show pressure which oscillates in time, the red line shows the primary acoustic force acting on particles located in the field, driving them to nodal planes of pressure (planes of minimal pressure fluctuation). The pressure field is critical as it facilitates the underlying physical mechanisms which are responsible for driving particles.

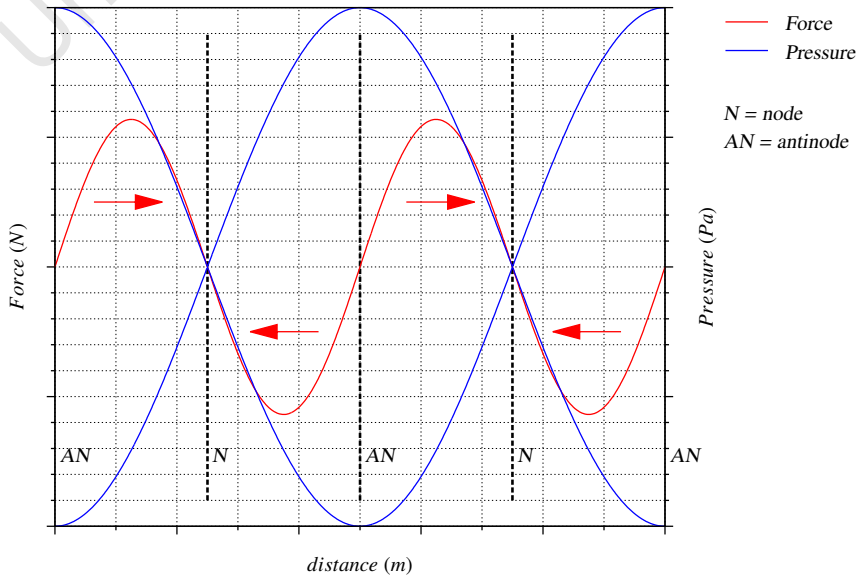


Fig. (1.3) *Acoustic force and pressure field in the fluid layer*

Three forces in general result from the application of acoustic fields in carrier media, two of which are intrinsic to the system and a third which arises from non uniformity of system constituents.

i. The primary acoustic force deflects particles toward nodal or anti-nodal pressure planes in a one dimensional system, such as that depicted in figure *1.3*. Pressure nodes correspond to positions within the carrier field where pressure does not fluctuate. Anti-nodal planes are those associated with pressure maxima and severe periodic fluctuation. Solid particles and liquid droplets generally tend toward nodal planes of pressure whilst gaseous bubbles tend toward anti-nodal planes to minimize the energy of the system

ii. A secondary acoustic force acts between particles. Between gas bubbles this force is more commonly known as the Bjerknes Force [51]. Depending on respective bubble resonant and acoustic driving frequencies bubbles may either attract, repel or have regimes of attraction and repulsion as functions of the inter particle separation distance. This force also acts between solid particles and between liquid droplets, but is governed by a different formulation

iii. The third force which may arise is known as the lateral force and may be attributed to a number of factors including reduction in energy density at wall boundaries, misalignment of reflector and transducer interfaces or from a combination of these factors [48, 51]. The lateral force is essentially an unintended component of the primary acoustic force which acts in planes perpendicular to that of the primary acoustic force

Counterproductive forces may become more pronounced when dealing with a fluid-acoustic-particulate system and deter the function of the acoustic force. These have to be considered in modelling and design applications. If precise particle manipulation in micro-chambers is of interest such as in bioreactors then acoustic streaming and thermally driven convection may become relevant. Brownian motion may also become pronounced when small particles are suspended in quiescent or thermally energized fluid. In larger filtration systems other forces may become dominant, such as those arising from the interaction between Kolmogorov or integral scale turbulent eddies and small particles. Forces such as buoyancy, convective drag, gravity, lift due to fluid velocity gradients as well as pressure gradient forces and others must be considered in order of their degree of relevance to a particular system.

Typical filtration systems make use of fluid induced forces such as drag and buoyancy, external forces such as gravity and centrifuge, alongside the previously listed acoustic forces to achieve the desired particle manipulations. It has been noted that even low particulate concentrations may affect the acoustic field and shift resonant frequency through scatter mechanisms. This effect on the system becomes more pronounced when particles aggregate within nodal or anti-nodal planes and often necessitates automated frequency control.

1.3 Applications

The acoustic force has been used in small and large scale applications, from multi-stage filtration systems to laboratory single particle manipulation and observation chambers. Typically small scale systems achieve filtration through particle deflection, whereas large scale systems use the acoustic

force to induce agglomeration of small particles, the resulting clumps are then more effectively filtered by subsequent conventional filtering. Conventional methods register poor filtration efficiencies when dealing with particles smaller than 2.5 micron [4]. Numerous researchers have noted the possibility to stage filtration segments in series to achieve even higher overall filtration efficiencies.

Small chambers fitted with acoustic transducers are used to facilitate non-contact positioning and manipulation operations on single particles. The acoustic force can be applied to inorganic and organic particles or cells as it does not cause any short term cell damage. Numerous papers deal with yeast filtration and red blood cell separation from plasma. Particles less than 1 micron in size have been manipulated, however the lower limit particle size which can be manipulated effectively is around 1 micron. Acoustic systems have been used to induce coalescence in emulsion flows containing liquid oil droplets in the size range 1 to 15 micron, the coalesced oil phase is then easily extracted from the carrier fluid using appropriate techniques [51].

The acoustic force has been used in conjunction with mesh filters i.e. pushing particles onto mesh fibres and releasing them when cleaning the system. Particles can be deflected from one immiscible fluid stream to another stream flowing alongside it, provided that the particle transfer does not produce undesired mixing of the fluids [48]. Possible areas of application include machining, powder densification, particle agglomeration and flocculation, drying and dewatering, etc, some of which have been commercially realized [8].

Systems implementing an acoustic force have many advantages over conventional counterparts. There are no moving parts, filters or membrane components which require maintenance and replacement. The technology can also be implemented under continuous flow operation [12]. Acoustic forces act on all particle types which has clear advantage over electrically or magnetically driven systems which depend on electric or magnetic particle properties. The acoustic force may be used to selectively deflect particles based on their density, compressibility or size differences.

1.4 Objectives

From the introductory text it is clear that the acoustic force has the ability to deflect particles, however to the authors knowledge no studies have been done investigating pure deflection of particles in large scale chambers of size relevant to industrial application. The aim of this thesis is to investigate particle deflection due to the primary acoustic force using computational means, in particular the objectives are:

- **i** To develop an extendible coupled CFD-DEM model incorporating the acoustic force using open source software OpenFOAM and LIGGGHTS
- **ii** To test the model on a case study investigating the deflection of sub 60 micron quartz particles in a square base duct section of dimension $50 \times 50 \times 70\ cm^3$
- **iii** To investigate the effect of frequency, particle size, particle node separation distance and flow rate on particle deflection efficiency

CHAPTER 2

BACKGROUND THEORY AND LITERATURE REVIEW

This chapter serves to introduce the reader to the relevant theory behind the acoustic force as well as provide an overview of the governing equations underlying the Computational Fluid Dynamics (CFD) and Discrete Element Method (DEM) algorithms, relating to the study undertaken. The second part of the chapter gives a broad overview of work which has been done in the field covering large and small scale applications as well as computational work.

2.1 Acoustic Theory

2.1.1 Resonant Cavities

Resonant cavities are relevant to the current study by virtue of the fact that standing waves are required to produce the desired pressure field within a fluid medium. Under the assumption of idealized wave chamber interaction i.e. neglecting dissipative effects of the fluid medium and non-uniformities of the actuator, equations *2.1.1 a* and *b* may be used to identify chamber resonant frequencies [31][58]. It should be noted that larger chambers allow for significant wave attenuation at exponential decay rates. Smith *et al* 1974, [31] shows an oscilloscope trace of an exponentially decaying wave form in a $61 \times 43 \times 24\ cm^3$ cavity under a *[1 0 0]* resonance mode.

$$f = \frac{c}{2}\sqrt{\left(\frac{n_x}{l_x}\right)^2 + \left(\frac{n_y}{l_y}\right)^2 + \left(\frac{n_z}{l_z}\right)^2} \qquad (2.1.1a)$$

$$f = \frac{cn_x}{2l_x} \qquad (2.1.1b)$$

Refer to the nomenclature list for symbol definition and associated units. Equation *2.1.1 a* reduces to equation *2.1.1 b* when considering planar waveforms and single mode resonances. The geometry must maintain pressure peaks at the boundaries to ensure a standing wave form is obtained, hence the wave must be in phase at the domain boundaries. This is physically consistent as the boundary pressures are expected to oscillate in unison, under the assumption of sonically hard walls to produce constructive interference, provided the chamber is dimensioned appropriately. Real standing wave forms may not be adequately described by the previous equation set, nevertheless standing waves can

be realized in finite sized chambers. An idealized assumption will be followed in the computational study.

2.1.2 Acoustic Fields in Fluids

The basic equations relevant to the study are those describing linear one dimensional spatial pressure, displacement and velocity distributions. The longitudinal perturbation is of sinusoidal form. The acoustic energy density ϵ_{ac} equation *2.1.10* is the defining characteristic of the primary acoustic force. It is proportional to the squares of driving frequency and driving amplitude. The acoustic energy density is derived from the sum of the potential and kinetic energies present within a fluid volume element and averaged with respect to time or space. Energy density is defined as a per volume quantity and contains pressure and velocity components which are associated with the potential and kinetic energies respectively, equations *2.1.2 a* and *b*. Equation *2.1.2 c* is the total energy [50].

$$\Delta E_k = \frac{1}{2}\,\rho_f\,u^2 V_0 \qquad (2.1.2a)$$

$$\Delta E_p = \frac{1}{2}\,\frac{p^2}{\rho_f\,c_f^2} V_0 \qquad (2.1.2b)$$

$$\frac{\Delta E}{V_0} = \frac{1}{2}\,\rho_f \left(u^2 + \frac{p^2}{\rho_f^2 c_f^2}\right) \qquad (2.1.2c)$$

Equation *2.1.2 c* is integrated with respect to period time or wavelength space to obtain the acoustic energy density form which is required to evaluate the primary acoustic force. The general one dimensional longitudinal pressure wave form is defined in equations *2.1.3 a* and *b* which accounts for waves travelling in opposite directions. If these propagating longitudinal waves are in phase and of equal magnitude they interfere constructively with one another and create a standing waveform in space which oscillates in time [50][59].

$$u(x,t) = -\omega A_+ \sin(\omega t - k_{ac}\,x) - \omega A_- \sin(\omega t + k_{ac} x) \qquad (2.1.3a)$$

$$p(x,t) = -\rho_f\,c_f\,\omega A_+ \sin(\omega t - k_{ac}\,x) + \rho_f\,c_f\,\omega A_- \sin(\omega t + k_{ac}\,x) \qquad (2.1.3b)$$

$$p(x,t) = 2\rho_f c_f \omega A_\pm \sin(k_{ac}\,x)\cos(\omega t) \qquad (2.1.3c)$$

The resulting standing waveform equation *2.1.3 c* has a sinusoidal spatially variant term $\sin(k_{ac}x)$ with double amplitude $2A$ and of oscillating frequency $\cos(\omega t)$. It is valid under the assumptions of ideal reflection and a non dissipative fluid medium. The acoustic force equation does in fact depend on spatial pressure distribution facilitating the need for a modified form of the force equation to account for non linearities. Detailed non-linear equations will not be included in this document as they are not relevant in the pursued context, however brief reference will be made as these considerations cannot be ignored in design work.

Typically acoustic field intensity is attenuated exponentially in a dissipative medium according to

Kinsler and Frey, 1962 [50]. Acoustic force on particles is proportional to the gradient of the time averaged acoustic field intensity which is related to acoustic energy density, equation *2.1.4 a* [47].

$$I = c_f \epsilon_{ac} \tag{2.1.4a}$$

$$I_x = I_0 e^{-2\alpha x} \tag{2.1.4b}$$

The attenuation coefficient α may be determined using equation *2.1.5 a* where two pressure measurements are sampled at successive pressure peaks along the planar acoustic axis in an experimental setup. There also exist analytic forms used to estimate the attenuation coefficient based on carrier medium properties, one such expression taking the form of equation *2.1.5 b*. Polar fluid media are more absorbent than non polar media [50].

$$P_2 = P_1 e^{-\alpha(x_2 - x_1)} \tag{2.1.5a}$$

$$\alpha = \frac{\omega^2}{2\rho_f \, c_f^3} \left(\frac{13}{3} \eta \right) \tag{2.1.5b}$$

Standing Wave Ratios (SWR)'s are determined experimentally or using a Smith Chart and are used to determine the ratio of incident to reflected wave amplitudes based on material, acoustic and geometric properties which in turn may assist in predicting more realistic non-linear pressure fields. In addition to the previous considerations, high concentrations of carried particulates and especially resonant bubbles may contribute to intensity attenuation and may have to be accounted for using modified forms of equation *2.1.5 b*. In addition to the contribution of particulates, attenuation from numerous sources may be summed into a single attenuation coefficient.

Appended table *B.1* documents measured properties of water which are required by equation forms quantifying the acoustic force. The carrier medium assumed in this study was water at around 20 °C. The tabulated entries were obtained from Kinsler and Frey's acoustics textbook [50].

2.1.3 Primary Acoustic Force

The Primary Acoustic Force is one of the central equations of the study and defines the effectiveness with which small particles of specified size range can be deflected. The following set of equations define the planar acoustic force, valid under the assumptions of lossless acoustic transmission through the carrier medium, perfect reflection at the wall boundaries and a one dimensional planar wave form. The following equations define the Primary Acoustic Force in two forms; one, the form typically found in literature *2.1.6 a* [48, 51][3, 15, 10, 37] and a reduced form *2.1.6 b* which reveals primary contributors. The framed equations complement equation *2.1.6 a* and define the outstanding variables in terms of more familiar quantities. Equation *2.1.6 b* was derived by substituting for all complex terms in equation *a*. The reduced form shows that force magnitude is dependant on driving frequency and driving amplitude as well as on particle size and particle node separation distance. The waveform is only affected by the driving frequency. Higher frequencies produce more nodes and anti-nodes i.e.

more closely spaced potential planes to which particles are forced.

$$F_{ac} = 4\,\pi\,k_{ac}\,\epsilon_{ac}\,a^3\,\Phi(\beta,\rho)\,\sin(2k_{ac}x) \tag{2.1.6a}$$

$$F_{ac} = A_{\pm}^2\,f^3\,a^3\,(2\pi)^4\,\frac{\rho_f}{c_f}\,\Phi\,\sin\left(\frac{4\pi f x}{c_f}\right) \tag{2.1.6b}$$

$$\epsilon_{ac} = \frac{\rho_f\,\omega^2 A_{\pm}^2}{2} \qquad k_{ac} = \frac{2\pi}{\lambda} \qquad \lambda = \frac{c_f}{f} \qquad \omega = 2\pi f$$

$$c_f^2 = \frac{\gamma\beta_T}{\rho_f} \qquad c_p^2 = \frac{E}{\rho_p} \qquad \beta_{p,f} = \frac{1}{\rho_{p,f}\,c_{p,f}^2}$$

The speed of sound equations are specific to the associated material and assume different forms for solid and liquid media. Different formulations must be used when dealing with liquid droplets and gas bubbles as opposed to solid particles based on respective bulk and elastic moduli. The acoustic contrast factor Φ equation *2.1.7* determines the sign of the acoustic force, which mathematically describes the physics of low density high compressibility particles being forced to pressure anti-nodes and high density low compressibility particles forced to pressure nodes minimizing the energy of the system.

$$\Phi = \frac{\rho_p + \frac{2}{3}\,(\rho_p - \rho_f)}{2\,\rho_p + \rho_f} - \frac{\beta_p}{3\,\beta_f} \tag{2.1.7}$$

Figure *2.1* illustrates the waveform overlap between the acoustic force and the associated ideal standing wave pressure field which will be assumed in this study. The magnitudes are somewhat unrelated as the pressure is a function of the driving amplitude and frequency whilst the acoustic force is a function of driving frequency and driving amplitude as well as particle size and other secondary parameters. Force is represented by the red line and pressure by the blue line seen to oscillate between $1e^4$ and $1e^{-4}$ in the hypothetical case. The pressure form is based on equation *2.1.3 c* and the acoustic force is based on equation *2.1.6 a*. The force acts on a 100 micron diameter particle. The transducer is set to a drive at 11835 Hz with vibrational displacement 50 micron. Nodal coordinates n_i are listed alongside the figure, obtained using the following expression.

$$x_{i0} = \frac{1}{k_{ac}}\left(n_i\pi + \frac{\pi}{2}\right) \qquad n_{1,2,3} \in 0, 1, 2... \tag{2.1.8}$$

The sinusoidal acoustic force has half the wavelength of the pressure wave which defines how particles will be forced. A half wavelength is equivalent to a doubled frequency and can be noted by examining equations *2.1.3 c* and *2.1.6*. The pressure form oscillates rapidly in time which effectively creates a quasi static field by virtue of its high frequency.

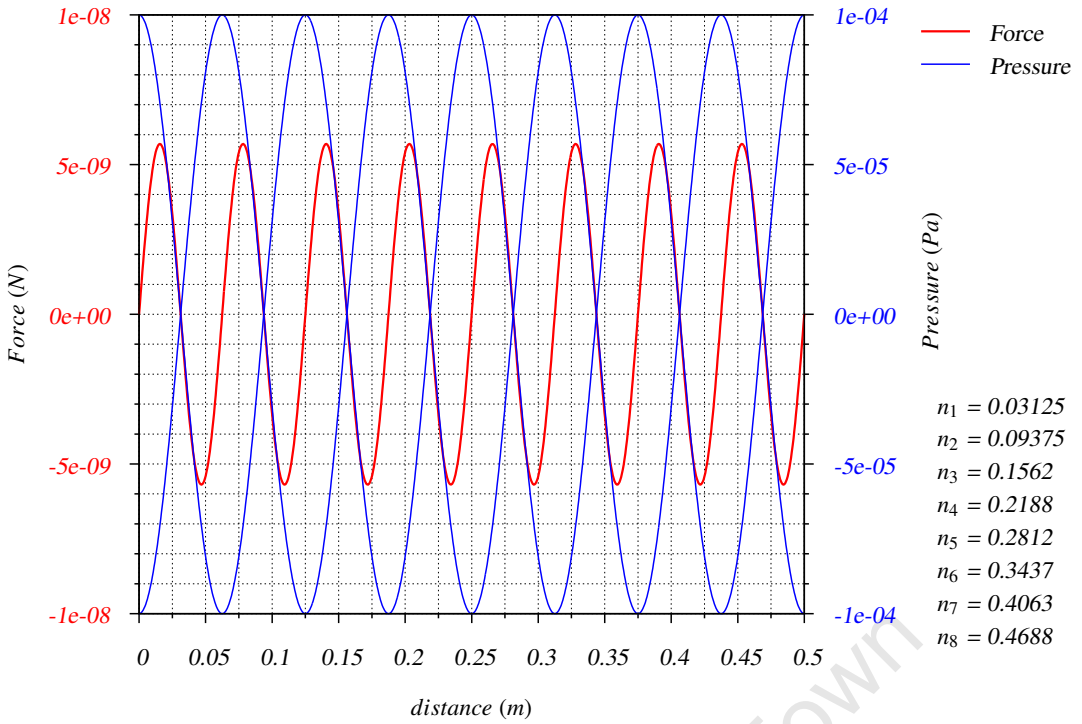


Fig. (2.1) *Acoustic force (left axis), oscillatory pressure (right axis) and pressure node positions* n_i, $A = 50\mu m$, $f = 11835Hz$, $r = 50\mu m$, $\rho_p = 2650kgm^{-3}$

The pressure form is used to represent regions of fluid compression and rarefaction. In the case of figure *2.1* the particles will be forced to the pressure nodes or pressure zeros. The particles are solid, of higher density and lower compressibility than the surrounding fluid medium and are as a result forced to positions of minimal pressure fluctuation to minimize the energy of the system.

An alternate expression, defining the acoustic force was derived by Gorkov to deal with arbitrary fields as opposed to the previous expression *2.1.6* which deals only with planar forms. Gorkov's derivation contains fluctuating time averaged pressure and velocity terms amidst fluid and particle properties. The formulation produces a contoured potential field in three dimensions which interacts with particles. The potential field $\mathbf{U_G}$ and its relation to the acoustic force is outlined in equations *2.1.9 a* and *b* [3]. Factors f_1 and f_2 define the sign of the acoustic force, their effect being equivalent to that of the acoustic contrast factor Φ defined previously.

$$\mathbf{U_G} = 2\pi a^3 \left[\left(\frac{\overline{\mathbf{p_{in}^2}}}{3\rho_f \, c_f^2} \right) f_1 - \left(\frac{\rho_f \, \overline{\mathbf{u_{in}^2}}}{2} \right) f_2 \right] \qquad (2.1.9a)$$

$$\mathbf{F_{ac}} = -\boldsymbol{\nabla} \mathbf{U_G} \qquad (2.1.9b)$$

$$\overline{\mathbf{p_{in}^2}} = \frac{1}{T} \int_0^T \mathbf{p}^2(\mathbf{x}, t) \, dt \qquad f_1 = 1 - \frac{\rho_f c_f^2}{\rho_p c_p^2}$$

$$\overline{\mathbf{u_{in}^2}} = \frac{1}{T} \int_0^T \mathbf{u}^2(\mathbf{x}, t) \, dt \qquad f_2 = \frac{2(\rho_p - \rho_f)}{(2\rho_p + \rho_f)}$$

The time averaged pressure and velocity fields are integrated with respect to the harmonic cycle period T and averaged over the magnitude of the same period. It should be noted that time or space integra-

tion with respect to wavelength yield the same result [50]. In addition to the previous consideration these fields are expressed as mean square fluctuations which is necessary to account for the oscillatory nature of the standing or travelling wave at a point where the particle is located. The contributions from the integrated mean squared time averaged pressure and velocity field fluctuations gives rise to the acoustic energy density ϵ_{ac} in its primitive form based on kinetic and potential energies outlined in equations *2.1.2 a-c*.

$$\epsilon_{ac} = \frac{1}{T}\int_0^T \left(\frac{p^2}{2\rho_f c_f^2} + \frac{\rho_f u^2}{2}\right) dt \qquad T = \frac{1}{f} \tag{2.1.10}$$

The harmonic period T is defined as the inverse of driving frequency. There is a clear resemblance between the potential field function derived by Gorkov, equation *2.1.9 a* and the acoustic energy density form of equation *2.1.10*. Both forms of the acoustic force, i.e. equation *2.1.6* or *2.1.9* assume that the particles located within the acoustic field have a much smaller radius than the prevalent wavelength i.e. $k_{ac}a \ll 1$ [3]. Equation *2.1.6 a* is recovered under plane wave conditions where the pressure and velocity forms are analytically described.

Secondary Acoustic Force

Secondary acoustic forces act between particles and are responsible in part for clustering and coalescence phenomena which are critical to understanding and simulating agglomeration. Equation *2.1.11* [51], is valid for solid particles and liquid droplets but not for gas bubbles.

$$F_{ac,2} = \frac{k_{ac}^2 \epsilon_{ac}}{2\pi}\left(1 - \frac{\beta_{p1}}{\beta_f}\right)\left(1 - \frac{\beta_{p2}}{\beta_f}\right)\frac{V_{p1}V_{p2}}{L_{12}} \tag{2.1.11}$$

Other inter-particle forces exist such as the van der Waals, double layer and hydrophobic forces amidst other material dependent forces such as charge forces, magnetic forces, etc. Forces between different types of particles have to be considered separately. Inter-particle forces typically become more significant as particles approach one another and can become relevant when simulating systems with high particulate concentrations where particle-particle interaction can determine flow evolution characteristics i.e. the carrier is coupled to the particles and the particles are coupled to one another. Dilute systems are effectively decoupled with respect to both particle fluid and particle-particle momentum exchange. Since the study deals with a dilute system secondary acoustic forces will not be accounted for in the simulation.

2.2 Modelling Theory

2.2.1 CFD

Navier Stokes Equations

CFD modelling is centred on solving the Navier-Stokes Equations (NSE), the incompressible continuity equation *2.2.1 a* and momentum equation *2.2.1 b* are given below respectively [40]. The assumption of fluid incompressibility allows the constant fluid density to be factored out of derivative terms and divided throughout. We assume incompressibility in the study on the basis that we are concerned more with the coupling system and particulate response to the acoustic force than with the exact flow solution. It is true that compressibility effects cannot be ignored when dealing with shock and sound wave propagation in fluid media [56], provided the flow and acoustic attenuation are of primary interest, in which case a second coefficient of viscosity denoted by the second Lamé constant λ, which has been the focus of much deliberation over the years, must be accounted for because $\boldsymbol{\nabla} \cdot \mathbf{U}$ is no longer zero. The second coefficient λ is related to bulk viscosity of the fluid which resists compression in much of the same way as a solid elastic material would and has been found to be around 3 times greater than shear viscosity [50]. The compressible momentum equations include terms $\delta_{ij}\,\lambda\,\boldsymbol{\nabla} \cdot \mathbf{U}$ which are nonzero in the principle directions. Since the model applies the acoustic force within the DEM environment, and is somewhat isolated from the CFD environment, and because we also assume a lossless propagation of sound, we neglect fluid compressibility for this ground level study.

$$\boldsymbol{\nabla} \cdot \mathbf{U} = 0 \tag{2.2.1a}$$

$$\frac{\partial \mathbf{U}}{\partial t} + \boldsymbol{\nabla} \cdot (\mathbf{U} \otimes \mathbf{U}) - \boldsymbol{\nabla} \cdot 2\nu\mathbf{D} = -\frac{1}{\rho}\boldsymbol{\nabla} p \tag{2.2.1b}$$

$$\mathbf{D} = \frac{1}{2}(\boldsymbol{\nabla}\mathbf{U} + \boldsymbol{\nabla}\mathbf{U}^T) \tag{2.2.1c}$$

Reynolds Averaged Navier Stokes

Turbulence modelling is a well researched and documented field, yet despite many years of turbulence related work the physics of turbulent behaviour are to this day not well understood. Many relevant equations are empirically based. Reynolds Averaged Navier Stokes (RANS) modelling, is based on averaging turbulent behaviour by decomposing the velocity field into average and fluctuating components according to equation *2.2.2 a* and introducing a Reynold Stress term to account for the fluctuating components as in equation *2.2.2 b*. Many turbulence models have been developed to compute the symmetric Reynolds Stress tensor $\mathbf{R}$, of those eddy viscosity models have been used most widely in engineering applications primarily because of their simplicity and computational efficiency. These models are based on the Bousinessq hypothesis which assumes the Reynolds Stress to be proportional to velocity gradients, and are characterized by the number of equations solved to compute the components of the Reynolds Stress tensor. Most if not all eddy viscosity models are based on the inaccurate assumption that turbulence is isotropic, and are unable to model secondary flows arising

from normal stress differences [35]. A fictitious turbulent viscosity is introduced to account for the added dissipative effect resulting from the net interaction of turbulent kinetic energy and dissipation and is used in part to formulate the Reynolds Stress tensor as in equation *2.2.2 c* [40].

$$\mathbf{U} = \overline{\mathbf{U}} + \mathbf{U}' \tag{2.2.2a}$$

$$\mathbf{R} = \overline{\mathbf{U}' \otimes \mathbf{U}'} \tag{2.2.2b}$$

$$\mathbf{R} = \nu_t \boldsymbol{\nabla} \mathbf{U} \tag{2.2.2c}$$

The NSE are modified to account for the averaged velocity field and Reynold Stress. The equations are solved for the averaged velocity and pressure fields. The resulting governing equations are outlined below *2.2.3 a*, *b* [40]. All barred quantities are averaged or evaluated using averaged components of velocity. An effective viscosity denoted by subscript e is introduced, defined as the sum of laminar and turbulent viscosities.

$$\boldsymbol{\nabla} \cdot \overline{\mathbf{U}} = 0 \tag{2.2.3a}$$

$$\frac{\partial \overline{\mathbf{U}}}{\partial t} + \boldsymbol{\nabla} \cdot (\overline{\mathbf{U}} \otimes \overline{\mathbf{U}}) + \boldsymbol{\nabla} \cdot \mathbf{R} - \boldsymbol{\nabla} \cdot 2\nu \overline{\mathbf{D}} = -\frac{1}{\rho} \boldsymbol{\nabla} \overline{p} \tag{2.2.3b}$$

$$\boldsymbol{\nabla} \cdot \mathbf{R} - \boldsymbol{\nabla} \cdot 2\nu \overline{\mathbf{D}} = -\boldsymbol{\nabla}^2 \left(\nu_e \overline{\mathbf{U}} \right) - \boldsymbol{\nabla} \cdot \left[\nu_e \, \mathbf{dev} \left(\boldsymbol{\nabla} \overline{\mathbf{U}}^T \right) \right] \tag{2.2.3c}$$

$$\mathbf{dev} \left(\boldsymbol{\nabla} \overline{\mathbf{U}}^T \right) = \boldsymbol{\nabla} \overline{\mathbf{U}}^T - \frac{1}{3} tr \boldsymbol{\nabla} \overline{\mathbf{U}}^T \tag{2.2.3d}$$

The previously mentioned eddy viscosity models cannot resolve vorticular structures nor can they account for secondary flows which occur in planes perpendicular to the primary flow direction as in the case of channelled duct flow. Secondary flows in ducts have been found to be around 1-2% of primary flow direction speeds [24]. Reynolds Stress Model (RSM)'s are preferable when simulating more complex physical flows with curvature, swirl, rotation and high strain rates. By virtue of the governing equations buoyancy, gravity and other forces are intrinsically accounted for. These models drop the assumption of isotropic turbulence and make use of six additional equations to solve for each component of the Reynolds Stress tensor, and are the recommended model for developing turbulent duct flow. They are however less stable than the eddy viscosity models and more difficult to implement. They are more computationally demanding than the simpler eddy viscosity models but less so than the other class of models in which Large Eddy Simulation (LES) falls, which retain much more of the physics through different averaging processes. LES models can be used to resolve turbulent eddies and vorticular flow behaviour of scale greater than a mesh determined filter size, however it is not as easily implemented as RANS type models and is more prone to numerical and convergence instability. LES is three dimensional and transient in nature, by virtue of the time dependent evolution and dissipation of turbulent eddies in three dimensional space, whereas RANS models may be used to model two dimensional flows.

There are numerous models available for computing the turbulent or eddy viscosity ν_t which is then used to evaluate the Reynolds tensor and subsequently to solve the RANS equations for velocity

and pressure. Velocity and pressure must be solved for simultaneously because they are coupled. RANS modelling, based on the Boussinesq hypothesis, justifies the use of a fictitious viscosity to bring about the same effect on the bulk flow as the combined effect of all turbulent eddies within the system. The previously defined RANS equations cannot be solved unless additional system equations are introduced. Some eddy viscosity type models available include additional $k-\epsilon$, $k-\omega$, Spalart Allmaras and others. Most are two equation models, some are pure algebraic models and others make use of a single equation. The model chosen for the current study was the $k-\epsilon$ two equation kinetic energy and dissipation rate model, motivated by the fact that this was a ground level study and flow complexities were sought to be avoided by using a simple and computationally efficient method at the expense of overall simulation accuracy. The additional $k-\epsilon$ equations follow in *2.2.4* [40]

$$P_k = 2\nu_t \left|\boldsymbol{\nabla}\overline{\mathbf{U}}\right|^2 \tag{2.2.4a}$$

$$\frac{\partial k}{\partial t} + \boldsymbol{\nabla}\cdot(\overline{\mathbf{U}}k) - \boldsymbol{\nabla}\cdot\frac{\nu_t}{\sigma_k}\boldsymbol{\nabla}k = P_k - \epsilon \tag{2.2.4b}$$

$$\frac{\partial \epsilon}{\partial t} + \boldsymbol{\nabla}\cdot(\overline{\mathbf{U}}\epsilon) - \boldsymbol{\nabla}\cdot\frac{\nu_t}{\sigma_\epsilon}\boldsymbol{\nabla}\epsilon = C_{1\epsilon}\frac{\epsilon}{k}P_k - C_{2\epsilon}\frac{\epsilon^2}{k} \tag{2.2.4c}$$

These equations must be solved for turbulent kinetic energy k and dissipation rate ϵ. The model coefficients have been determined empirically to suite the widest range of flows and should not have to be adjusted for general applications. An initialized non-zero kinetic energy field is required as well as a dissipation rate field which may or may not be null. The CFD software Open source Field Operation and Manipulation (OpenFOAM) used in the study produces the turbulent viscosity field using the initialized fields. OpenFOAM and FLUENT document contrasting equations relating to the estimation of kinetic energy, dissipation rate and turbulent viscosity, based on length scales and turbulent intensity, required for initialization. FLUENT documentation [45] was used as the equation basis for initializing the kinetic and dissipation rate fields.

OpenFOAM suggests these quantities be estimated as percentages of the characteristic flow velocity. An OpenFOAM user manual [43] example assumes initial isotropic turbulence at 5% of the characteristic flow velocity and the length scale at 20% of the characteristic velocity. For two equation models turbulent length and intensity measures are used to evaluate kinetic energy and dissipation rate which are subsequently required in part to initialize the system. Equations *2.2.5 a* and *b* are used to produce initial values for the OpenFOAM numerical solver based on FLUENT documentation [45]. Once k and ϵ are specified the solver computes turbulent viscosity based on equation *2.2.5 c*.

$$k = \frac{3}{2}(\bar{u}I_t)^2 \tag{2.2.5a}$$

$$\epsilon = C_\mu^{\frac{3}{4}}\frac{k^{\frac{3}{2}}}{\ell} \tag{2.2.5b}$$

$$\nu_t = C_\mu\frac{k^2}{\epsilon} \tag{2.2.5c}$$

$$I_t \equiv \frac{u'}{\bar{u}} = 0.16(Re_{D_H})^{-\frac{1}{8}} \qquad D_H = \frac{4A_c}{P_c}$$

$$\ell = 0.07 D_H \qquad Re_{D_H} = \frac{L_H \left|U\right|}{\nu}$$

The boundary layer is typically defined as that distance normal to the wall or plate to a point where the flow velocity has reached 99% of its freestream value. A freestream value is the average velocity of a flow profile. It is expected to be larger for turbulent plug flows than for laminar parabolic flows, however the velocity gradient at turbulent boundaries is large hence suggesting a narrow boundary layer thickness in turbulent flows.

The near wall velocity profile has three distinct regions namely the viscous sublayer, the buffer or transition layer and the logarithmic layer. Equation *2.2.6 a* is used to define the dimensionless velocity profile relation in the viscous sublayer whilst equation *2.2.6 b* is valid in the logarithmic layer. There is no analytic form describing the buffer layer.

$$u^+ = y^+ \qquad y^+ < 5 \tag{2.2.6a}$$

$$u^+ = \frac{1}{\kappa} ln(y^+) + B^+ \qquad y^+ > 30 \tag{2.2.6b}$$

$$u^+ = \frac{u}{u_\tau} \qquad y^+ = \frac{u_\tau}{\nu} y_{c1} \tag{2.2.6c}$$

$$u_\tau = \sqrt{\frac{\tau_w}{\rho_f}} \qquad \tau_w = \mu_e \left.\frac{du}{dy}\right|_{y=0}$$

Wall functions make use of these analytic but discontinuous forms to describe the flow between the wall and the wall adjacent cell centre. They facilitate the use of larger wall adjacent mesh cells, to optimize computational time, however these cells have to be sized carefully so that the patching between the wall and the centre of the wall adjacent cell does not fall into the transition region between the viscous sublayer and the logarithmic layer. When using log of the wall patch functions a y^+ parameter, which essentially relates velocity profile to mesh, must be maintained within certain ranges. The y^+ parameter can only be computed during a simulation run as it is solution and mesh dependent, hence the mesh has to be adjusted iteratively based on the solution. The upper limit of y^+ varies somewhat between 150 and 300. A y^+ which falls within the buffer region i.e. $5 < y^+ < 30$ should be avoided as the patching function cannot define this region analytically and the bulk flow will be affected by the misplaced patching. OpenFOAM has in-house functions which may be used to evaluate y^+ for RANS and Large Eddy simulations. The LES associated utility is based on the standard definition of y^+ whereas the RANS utility evaluates a quantity known as y^* which is defined in equation *2.2.7* and is comparable to the standard definition of y^+ in the logarithmic layer.

$$y^* = \frac{C_\mu^{0.25} k^{0.5} y_{c1}}{\nu} \tag{2.2.7}$$

The Courant number Co equation *2.2.8* defines the ratio of incremental solution propagation distance to cell size in the flow direction and is a measure of the adequacy of solution discretization. A Courant

number field is generated during simulation from which the maxima can be extracted. The Courant number should theoretically be maintained below unity to ensure that flow propagation is adequately captured within a single cell length and that the solution does not skip cells. This is not always achievable in transient simulation because the flow velocity evolves over time and can subsequently inflate the Courant number. Numerical instability, inaccuracy and divergence can result if the solution propagates across several cells during a single time step. LES is particularly sensitive to Courant number which should in certain cases be reduced to 0.5 or lower. Solver improvements over the years have introduced additional numerical stability at higher Courant Numbers. An Adaptive Time Stepping (ATS) code based on maintaining Courant number at a user specified level is implemented in certain OpenFOAM solvers. The number is of particular importance in multiphase flows where either phase can limit the time step according to the Courant number criterion of that phase. The ATS code was introduced into the standard incompressible Pressure-Implicit Split-Operator (PISO) solver for this study, the implementation of which is detailed in section *3.2.2*.

$$Co = \frac{\delta t\,|\mathbf{U}|}{\delta x} \tag{2.2.8}$$

A particularly useful equation *2.2.9* pertaining to turbulent pipe and duct flows is that derived by Weisbach [55]. Known as the Darcy-Weisbach equation, it relates fluid pressure drop across a given length of pipe or duct of any cross section, to the average flow velocity through a dimensionless friction factor proposed by Darcy. The relation is valid for laminar and turbulent flows where the friction factor is obtained in relation to the prevalent Reynolds number from the Moody chart.

$$\Delta P = f_D \frac{L}{D_H} \frac{\rho_f V_m^2}{2} \tag{2.2.9}$$

In addition to the Darcy-Weisbach equation another expression, equation *2.2.10* may be relevant in relating friction factor to the ratio of mean and maximum flow velocities in turbulent flows [55]. Equations *2.2.9* and *2.2.10* may be used in part to give some validation to simulation results.

$$\frac{V_m}{V_{max}} = \left(1 + 1.33\sqrt{f_D}\right)^{-1} \tag{2.2.10}$$

PISO

The PISO algorithm solves the NSE for velocity and pressure through a predictor corrector algorithm. All fluid velocity related terms are grouped on the Left Hand Side (LHS) of equation *2.2.3 b* and all remaining terms on the Right Hand Side (RHS) of the NSE in order to generate the required coefficient matrix and force vector to solver the non-linear NSE. Equation *2.2.3 b* does not account for additional forcing terms aside from pressure gradient on the RHS, however these may be included if necessary. It should be noted that all non-linear problems have coefficient matrices which depend on the solution variable itself, in this case velocity, hence the coefficient matrix varies with the solution variable. Because of the coupled nature of fluid dynamics problems with respect to pressure and velocity, both variable fields have to be solved by advancing and correcting the solution iteratively.

The velocity field is predicted initially using an estimated pressure field or the initialized pressure field. The pressure field is then corrected based on a Poisson pressure correction equation which arises under the condition of incompressibility. A flux type equation form is generated on which mass continuity is enforced through a zero divergence condition. The NSE are decomposed into a velocity related coefficient matrix $\mathbf{C}$, a pressure gradient related vector $\boldsymbol{\nabla} p$ and a source term vector expression $\mathbf{r}$ which encompasses all other momentum sources. It is assumed that these momentum sources are independent of the velocity and pressure fields themselves. The resulting discretized momentum equation *2.2.11 a* results. Furthermore the $\mathbf{C}$ matrix may be decomposed into its diagonal $\mathbf{A}$ and off diagonal $\mathbf{H}'$ components respectively as in equation *2.2.11 b*. A newly defined $\mathbf{H}$ operator equation *2.2.11 d* encompasses both source terms and previously computed velocity fields. The initial velocity prediction step is represented mathematically by equation *2.2.11 e* where the velocity subscript 0 denotes the first velocity prediction and the pressure subscript $0, n$ denotes the initial pressure guess n for the zeroth pressure iteration corresponding to the zeroth velocity. The second velocity iteration equation *2.2.11 i* denoted by subscript 1 may be expressed as a function of the previous velocity field 0 through the $\mathbf{H}$ operator and the first corrected pressure $n + 1$, 1 being the number of pressure iterations specified by the user and defaulted to one. The pressure is not known at the first velocity iteration hence the requirement for pressure correction by generating a surface flux ϕ equation *2.2.11 f* to which we can apply the zero divergence condition equation *2.2.11 g*. All surface flux quantities are denoted by subscript f. The resulting equation *2.2.11 h* is solved to obtain the iterated pressure $n + 1$. The zero pressure subscript remains as it is the corrected pressure corresponding to the zeroth velocity iteration, which is inherent in the flux itself through the $\mathbf{H}$ operator. The pressure solution may be iterated more than once by correcting the flux field iteratively. Once the specified number of pressure iterations and flux corrections have been performed the subsequent velocity iteration can be computed as in equation *2.2.11 i* using the corrected pressure [57]. Once the velocity has been pressure corrected a new $\mathbf{H}$ matrix can be computed using equation *2.2.11 j*. Using the updated $\mathbf{H}$ matrix the flux equation is modified as in equation *2.2.11 l* and the pressure correction can begin again, this time corresponding to the first velocity iteration denoted by subscript 1. The last computed pressure is carried into the flux equation as in *2.2.11 k*. A number of pressure corrected velocity iterations, as specified by the user, may be carried out but is defaulted to 2. Velocity corrections are referred to as orthogonal corrections and pressure corrections are referred to as non-orthogonal corrections.

$$\mathbf{C}\mathbf{u}_n = \mathbf{r} - \boldsymbol{\nabla} p \tag{2.2.11a}$$

$$\mathbf{A} + \mathbf{H}' = \mathbf{C} \tag{2.2.11b}$$

$$\mathbf{A}\mathbf{u}_0 + \mathbf{H}'\mathbf{u}_0 = \mathbf{r} - \boldsymbol{\nabla} p_{0,n} \tag{2.2.11c}$$

$$\mathbf{H}_0(\mathbf{u}_0) = \mathbf{r} - \mathbf{H}'\mathbf{u}_0 \tag{2.2.11d}$$

$$\mathbf{u}_0 = \mathbf{A}^{-1}\mathbf{H}_0(\mathbf{u}_0) - \mathbf{A}^{-1}\boldsymbol{\nabla} p_{0,n} \tag{2.2.11e}$$

$$\phi_{0,n} = \left[\mathbf{A}^{-1}\mathbf{H}_0(\mathbf{u}_0)\right]_f \cdot \mathbf{S}_f - \left(\mathbf{A}_f^{-1}\boldsymbol{\nabla}_f\, p_{0,n}\right) \tag{2.2.11f}$$

$$\boldsymbol{\nabla} \cdot \phi_{0,n} = 0 \tag{2.2.11g}$$

$$\boldsymbol{\nabla} \cdot \left(\mathbf{A}_f^{-1} \boldsymbol{\nabla}_f \, p_{0,n+1} \right) = \boldsymbol{\nabla} \cdot \left[\mathbf{A}^{-1} \mathbf{H}_0(\mathbf{u}_0) \right]_f \cdot \mathbf{S}_f \tag{2.2.11h}$$

$$\mathbf{u}_1 = \mathbf{A}^{-1} \mathbf{H}_0(\mathbf{u}_0) - \mathbf{A}^{-1} \boldsymbol{\nabla} p_{0,n+1} \tag{2.2.11i}$$

$$\mathbf{H}_1(\mathbf{u}_1) = \mathbf{r} - \mathbf{H}' \mathbf{u}_1 \tag{2.2.11j}$$

$$p_{1,n} = p_{0,n+1} \tag{2.2.11k}$$

$$\phi_{1,n} = \left[\mathbf{A}^{-1} \mathbf{H}_1(\mathbf{u}_1) \right]_f \cdot \mathbf{S}_f - \left(\mathbf{A}_f^{-1} \boldsymbol{\nabla}_f \, p_{1,n} \right) \tag{2.2.11l}$$

2.2.2 DEM

Background and Time Integration

The Discrete Element Method was introduced by Cundall and Strack in the late 1970's [25, 20]. The method accounted for individual particle trajectories in Lagrangian frames of reference. The method has proven to be an invaluable research tool from the time of its introduction to the present day and will continue to be improved upon as more computational power becomes available to the researcher. Essentially DEM allows bottom up research i.e. investigations of how micro mechanisms and interactions translate into macro observable effects instead of the more common top down type research which focusses only on continuum macroscopic phenomena. Three typical particle collision models exist, hard and soft sphere models for rigid particle interactions, of which the soft sphere is implemented in classical DEM code, and Finite Discrete Element Method which may be used to simulate deformable particles. All three methods account for dissipated energy during collision resulting in smaller rebounding velocities compared with incident velocities. Static or equilibrium state particulate systems cannot be modelled using hard sphere models whereas soft sphere models can be used. Simulation accuracy and computational effort increase from hard to soft to finite element models [21].

Trajectories are computed by solving translational equation *2.2.15 a*, and angular momentum balance equations which will not be detailed in this study as sphere rotation is not expected to contribute significantly to the motion of particles. Particle contact, cohesion, rolling friction and collision models on which DEM modelling in LIGGGHTS (the proposed DEM modelling software, refer to section *3.1.3* if an overview is needed at this point) is based, are derived from Hertzian non linear and Hookean linear collision theories, of which the Hertzian formulation was adopted in this study. A soft sphere spring dashpot model is used with normal and tangential stiffness and damping, based on particle overlap and relative normal and tangential velocities. Particle overlap is generally limited to $< 0.5\%$ of particle diameter. Soft sphere models are preferred over hard sphere models because of their suitability toward modelling multiple sphere contacts as well as time dependant effects. All particles are approximated as smooth spheres, however LAMMPS Improved for General Granular and Granular Heat Transfer Simulations (LIGGGHTS) code can be used to model spherical and non-spherical particles. Rugged forms are approximated by clumping spherical particles together. Cohesion is based on Hertzian theory. Various contact schemes may be selected such as those which include or exclude tangential

damping, rolling friction and cohesion. If cohesion is flagged an additional attractive normal force between contacting spheres is applied.

An interject is made here to focus the reader's attention on the fact that OpenFOAM (the proposed CFD modelling software, refer to section *3.1.3* if an overview is needed at this point) has in-house libraries dedicated to Lagrangian particle tracking, which have been used successfully to model sprays and solid particles involving particle-particle and particle-wall collisions, however the collision mechanics are less involved than those implemented in LIGGGHTS. A hard contact model is used implementing a coefficient of restitution to compute post contact velocities [60] based on impact momentum exchange. In addition only a fluid drag force is incorporated as standard whereas CFDEM (the coupling code built as an extension to OpenFOAM and tailored for LIGGGHTS) allows for a number of fluid induced forces to be applied without additional programming. OpenFOAM contact detection models are based on cell encapsulation and not on specific particle separation distances. In general the framework is suited more for dilute systems where contact is of secondary importance and the particle phase does not feed back on the flow. It is true that the model dealt with in this study is dilute however the possibility of extension is better facilitated through implementation of more heavy duty DEM code.

Large-scale Atomic Molecular Massively Parallel Simulator (LAMMPS) (base code on top of which LIGGGHTS was built, section *3.1.3*) is primarily designed for parallel processing to harness and combine computational resources, hence an integral part of the internal workings of LAMMPS and LIGGGHTS is devoted to its parallel communication between processors. The implemented spatial decomposition method allows for efficient cross processor data transfer. The idea behind spatial decomposition is to allocate sub domains along with their containing particles or atoms to the available processors, one sub domain and all its particles to one processor. Information relating to contained atoms is complete, whilst potentially interacting atoms or particles in the vicinity of the sub domain are tracked by an incomplete set of data i.e. only data required to compute bond or collision forces between external 'ghost' atoms and processor 'owned' atoms, is stored on the respective processor. 'Ghost' atom data is communicated and updated between processors throughout the simulation run. Each particle or atom has an associated neighbour list which is built as necessary to account for nearby particles.

Velocity Verlet integration was implemented in this study. The algorithm is detailed in equations *2.2.12 a-d* [28]. The corresponding LAMMPS code fragment *C.1* was obtained from presentation slides used in a lecture given by C. Kloss, the core developer of LIGGGHTS. The algorithm updates particle velocity and position, computing a full timestep displacement advance *2.2.12 a* and a half timestep velocity advance *2.2.12 b* using velocity and acceleration from a previously computed timestep or initialized values. The acceleration term in equation *2.2.12 c* is computed by solving Newton's equation of motion. The last step *2.2.12 d* involves completing the velocity integration by advancing the remaining half step. Verlet integration position error is of the order $\sim\Delta t^4$.

$$\mathbf{x}(t+\Delta t) = \mathbf{x}(t) + \mathbf{v}(t)\Delta t + \frac{1}{2}\mathbf{a}(\Delta t)^2 \quad (2.2.12a)$$

$$\mathbf{v}(t+\frac{\Delta t}{2}) = \mathbf{v}(t) + \frac{\mathbf{a}(t)\Delta t}{2} \quad (2.2.12b)$$

$$\mathbf{a}(t+\Delta t) = \frac{\sum_k \mathbf{F}_k}{m_p} \quad (2.2.12c)$$

$$\mathbf{v}(t+\Delta t) = \mathbf{v}(t+\frac{\Delta t}{2}) + \frac{\mathbf{a}(t+\Delta t)\Delta t}{2} \quad (2.2.12d)$$

A single processor performs these advances only on its 'owned' atoms. The updated positions are then communicated amongst processors from which neighbour lists may be updated and atom data may be migrated from one processor to the next if the particle has moved into an adjacent sub domain. Neighbour lists are built as necessary every n timesteps, computed through separate dedicated algorithms which will not be discussed here. Based on the new discrete configuration all relevant bonding and non bonding forces may be computed as many of these are functions of particle separation distance. Any partial forces which have been computed between particles residing in different domains are communicated amongst processors. It should be noted that a global particle index track is maintained to facilitate particle migration between processors [28].

LAMMPS implements a background grid of optimized size which is used to locate particles and identify potential contact pairs. Each cell of the grid is referred to as a bin and contains a number of particles or no particles. The grid based algorithm implemented in LAMMPS reduces computation time by excluding potential partners and distant neighbours, and since the bulk of DEM computation time is devoted to computing particle interactions the additional code complexity is justified. Neighbouring particles are detected by traversing grid elements left to right, and bottom up identifying all particles with centres within cell geometric grid boundaries, and then identifying all neighbouring particles located within immediately adjacent cells to the right and below the current cell. Identified pairs are written to a neighbour list if the distance between their centres is below a cut-off length. Collision detection is then based on the generated neighbour list for the next N_v steps, where N_v is a function of time step, maximum relative velocity and a parameter used in determining critical length [21].

Forces

The normal force component of collision equation *2.2.13 a*, from a computational point of view, is a function of overlap between two contacting particles and their relative normal velocity, whilst the tangential component of force equation *2.2.14 a* has a similar formulation, accumulating with contact time and limited by Coulombs frictional limit equation *2.2.14 b* [20, 25][52][61]. The normal and tangential stiffness and damping coefficients $k_{n,t}$ and $c_{n,t}$ respectively are computed using material property data and derived quantities which define effective contact characteristics. Instead of specifying material properties, stiffness and damping coefficients may be entered directly if certain contact characteristics are desired, or if such data is more readily available through testing or otherwise. Hertzian non-linear contact potentials have been adopted in this study and are detailed in equations *2.2.13 b* and *c* and *2.2.14 c* and *d* supplementing the respective force equations [61]. The inclusion of rolling friction

would serve to add an additional torque contribution and would affect particle rotation. The normal contact Herzian model equation *2.2.13 a* contains an additional third term contribution resulting from a cohesion force which may or may not be included at the users discretion. The cohesion force is proportional to the interfacial contact area by a user specified cohesion energy density coefficient.

$$\mathbf{F}_n = -k_n \delta \mathbf{x}_{p,n} + c_n \mathbf{u}_{p,n} + \rho_E \mathbf{A}_n \tag{2.2.13a}$$

$$k_n = \frac{4}{3} Y^* \sqrt{R^* \delta_n} \tag{2.2.13b}$$

$$c_n = -2\sqrt{\frac{5}{6}}\beta\sqrt{\frac{3}{2}k_n m^*} \geq 0 \tag{2.2.13c}$$

$$\frac{1}{m^*} = \frac{1}{m_1} + \frac{1}{m_2} \qquad \delta_n = R_1 + R_2 - r \qquad \frac{1}{Y^*} = \frac{(1-\nu_1^2)}{Y_1} + \frac{(1-\nu_2^2)}{Y_2}$$
$$\frac{1}{R^*} = \frac{1}{R_1} + \frac{1}{R_2} \qquad \beta = \frac{\ln(e)}{\sqrt{\ln^2(e) + \pi^2}}$$

The incremental tangential displacement vector between contacting particles, equation *2.2.14 a* is obtained through the time integral of relative tangential velocity truncated to satisfy the frictional yield criterion [61]. The tangential force equation *2.2.14 a* is a function of contact time through the time integral, which acts as an incremental spring that stores energy from relative tangential motion [20]. The tangential force also contains a velocity damping component which dissipates contact energy and may or may not be included at the users discretion. The tangential force continues to increase until the frictional limit has been reached, thereafter the limiting tangential force is adopted and remains in effect until the particles separate or until tangential force falls below the limit once more.

$$\mathbf{F}_t = \min\left\{ \left| k_t \int_{t_{c,0}}^{t} \mathbf{u}_{p,t}\, dt + c_t \mathbf{u}_{p,t} \right|, \mu_c \mathbf{F}_n \right\} \tag{2.2.14a}$$

$$\mathbf{F}_t \leq \mu_c \mathbf{F}_n \tag{2.2.14b}$$

$$k_t = 8G^* \sqrt{R^* \delta_n} \tag{2.2.14c}$$

$$c_t = -2\sqrt{\frac{5}{6}}\beta\sqrt{k_t\, m^*} \geq 0 \tag{2.2.14d}$$

$$\frac{1}{G^*} = \frac{2(2+\nu_1)(1-\nu_1)}{Y_1} + \frac{2(2+\nu_2)(1-\nu_2)}{Y_2}$$

The general equation of motion for a single particle may be expressed as in equation *2.2.15 a* [20, 25][44, 45] where the forces on the RHS of the equation are in order and by subscript the drag force $\mathbf{F}_D$ [21][44, 45], gravity buoyancy force $\mathbf{F}_{GB}$ [21][44, 45], virtual mass force also known as the carried or added mass force $\mathbf{F}_V$ [21][44, 45], pressure gradient force $\mathbf{F}_P$ [21][44, 45], Basset force $\mathbf{F}_B$ *2.2.16 a* [23][44], Saffman force and Magnus force which are combined into a single lift force term $\mathbf{F}_{SM}$ *2.2.17 a* [23, 21], collision force $\mathbf{F}_C$ which is comprised of the normal $\mathbf{F}_n$ equation *2.2.13 a* and tangential $\mathbf{F}_t$

equation *2.2.14 a* force components, and other forces $\mathbf{F}_{...}$ including magnetic, electrostatic, Brownian, thermophoretic, etc which will not be detailed in this review. In certain cases depending on particle size and material properties as well as flow characteristics, a number of the forces mentioned can be neglected. The framed compliments of equation *2.2.15 a* are implemented as documented here in the coupling code, under the assumption of their validity over a wider range of Reynolds numbers.

The drag force may be attributed to a pressure gradient which manifests due to fluid particle velocity slip as well as viscous shears on the particle surface. The drag coefficient C_D is sensitive to particle Reynolds number, particle shape, size, material properties, adjacent particles, and can take on a number of different forms during particle travel depending on variations in local flow characteristics. The specific form of the drag coefficient used in the study will be detailed in section *2.2.3*. The virtual mass force ties in with the Basset force, both of which arise in accelerating flow environments. The virtual mass force is associated with accelerating the fluid surrounding a particle and may become relevant when particle density is less than that of the surrounding fluid [44, 45]. The pressure gradient force is self describing and may become relevant in the presence of shock and transverse pressure gradient flows.

$$m_p \frac{d\mathbf{U}_p}{dt} = \mathbf{F}_D + \mathbf{F}_{GB} + \mathbf{F}_V + \mathbf{F}_P + \mathbf{F}_B + \mathbf{F}_S + \mathbf{F}_M + \mathbf{F}_C + \mathbf{F}_{...} \qquad (2.2.15a)$$

$$\mathbf{F}_D = \frac{3C_D}{4d_p}\frac{m_p\,\rho_f}{\rho_p}|\mathbf{U}_f - \mathbf{U}_p|(\mathbf{U}_f - \mathbf{U}_p) \qquad \mathbf{F}_{GB} = \left(\frac{\rho_f}{\rho_p} - 1\right) m_p \mathbf{g}$$

$$\mathbf{F}_V = \frac{1}{2}\frac{m_p\,\rho_f}{\rho_p}\frac{d}{dt}(\mathbf{U}_f - \mathbf{U}_p) \qquad \mathbf{F}_P = -\frac{m_p}{\rho_p}\boldsymbol{\nabla} p$$

The Basset force accounts for accumulated particle accelerations and decelerations and is essentially the unsteady component of drag force [23]. It may become significant when a particle is accelerated at a high rate or subject to intermittent 'jerks' and less so when the fluid particle density ratio is small [44]. To the authors knowledge the Basset force has not been included in coupling code, but is documented in equation *2.2.16 a* for completeness. The constants, c_1 and c_2 are empirical.

$$\mathbf{F}_B = 18\,\frac{m_p\,\mu_f}{\rho_p d_p^2}\int_{-\infty}^{t} K(t-\tau)\frac{d}{d\tau}(\mathbf{U}_f - \mathbf{U}_p)d\tau \qquad (2.2.16a)$$

$$K(t-\tau) = \left(\left[\frac{4\pi\nu(t-\tau)}{d_p^2}\right]^{\frac{1}{2c_1}} + \left[\frac{\pi(t-\tau)^2}{f_H\tau_d}Re_p^3\right]^{\frac{1}{c_1}}\right)^{-c_1}$$

$$Re_p = \frac{|\mathbf{U}_f - \mathbf{U}_p|d_p}{\nu} \qquad f_H = (0.75 + c_2 Re_p)^3 \qquad \tau_d = \frac{d_p^2}{\nu}$$

Continuous phase vorticity and particle rotation have been identified as the two primary mechanisms attributing to particle lift phenomena [23]. The lift forces of Magnus and Saffman based on particle rotation and shearing velocity gradients across the particle respectively, have been combined to produce the set of equations *2.2.17 a* and its compliments. The integration of the two equations is valid under

'free rotation' which assumes zero particle torque. Another assumption which must be made is that spin and shear are both perpendicular to relative velocity. The formulation as presented in this document is implemented in the coupling code CFDEM under both assumptions. Saffman lift force may be attributed to a pressure gradient which is set up across a sphere because of a difference in flow speed at opposite sides of the particle [23]. The velocity shear induced lift force may become relevant near wall boundaries where velocity gradients are high, and less so under low shear rates and low particle Reynolds numbers [44]. The Magnus lift force may also be attributed to a pressure gradient which asserts itself across a rotating sphere as a result of fluid entrainment on one side. Lift induced through particle rotation becomes negligible compared with drag when the particle is small or when spin velocity is low [44].

$$\mathbf{F}_{SM} = \frac{3}{4}\frac{m_p\,\rho_f}{\rho_p\,d_p}C_L|\mathbf{U}_r|\frac{\mathbf{U}_r\times\boldsymbol{\omega}}{|\boldsymbol{\omega}|} \tag{2.2.17a}$$

$$C_L = J^*\frac{12.92}{\pi}\sqrt{\frac{\omega^*}{Re_p}} + \Omega^*_{p,eq}C^*_{L,\Omega}$$

$$C^*_{L,\Omega} = 1 - \left\{0.675 + 0.15\left(1+\tanh\left[0.28\left(\frac{\omega^*}{2}-2\right)\right]\right)\right\}\tanh\left[0.18\sqrt{Re_p}\right]$$

$$J^* = 0.3\left\{1+\tanh\left[2.5\left(log_{10}\left(\sqrt{\frac{\omega^*}{Re_p}}\right)+0.191\right)\right]\right\}\left\{\frac{2}{3}+\tanh\left[6\sqrt{\frac{\omega^*}{Re_p}}-1.92\right]\right\}$$

$$\Omega^*_{p,eq} = \frac{\omega^*}{2}\left(1-0.0075Re_\omega\right)\left(1-0.062\sqrt{Re_p}-0.001Re_p\right)$$

$$Re_\omega = \frac{|\boldsymbol{\omega}|d_p^2}{\nu} \qquad \omega^* = \frac{|\boldsymbol{\omega}|d_p}{|\mathbf{U}_r|} \qquad \mathbf{U}_r = \mathbf{U}_f - \mathbf{U}_p \qquad \boldsymbol{\omega} = \boldsymbol{\nabla}\times\mathbf{U}_f$$

Time Step

Adequate time step selection in any DEM simulation is key to resolving discrete phase collisions either between particles or between particles and bounding surfaces. Two time based criteria, Hertz and Rayleigh, can be called to give the user an indication as to the appropriateness of the selected time step. If the timestep is chosen too large the energy propagation of the system would be erroneously computed. Particle overlap represents energy stored in particle deformation and since it is limited to $< 0.5\%$ of the particle diameter, excessive overlap brought about under large time stepping would result in non-physical energy generation. Energy transmission in granular materials has been primarily attributed to Rayleigh surface waves which account for some 67% of total energy, followed by transverse body waves accounting for another 26%. For computational purposes it is assumed that Rayleigh transmission is total. The difference in transmission speed between Rayleigh and transverse waves is small hence the assumption can be made. Rayleigh time is that time taken by the Rayleigh surface wave to propagate through a particle and is described by equation *2.2.18 a* [22, 21].

$$T_R = \pi R\left(\frac{\rho_p}{G}\right)^{\frac{1}{2}}(0.1631\nu + 0.8766) \tag{2.2.18a}$$

$$G = \frac{Y}{2(\nu + 1)} \tag{2.2.18b}$$

It is recommended to use a simulation time step anywhere between 10 and 30% of Rayleigh time. Particle contact time is based on Hertzian collision theory equation *2.2.19*. The recommended simulation time step required to resolve particle contact is anywhere between 5 and 15% of the Hertzian time measure [21]. The relative velocity v_r in equation *2.2.19* refers to the maximum relative velocity between two potentially colliding particles or between a particle and a surface. The code implementation of the Hertzian time equation is somewhat modified in terms of accounting for hypothetical particle collision with itself. The code also accounts for moving surface geometry in computing the maximum relative velocity.

$$T_H = 2.87 \left(\frac{m^*}{R^* Y^{*2} v_r} \right)^{\frac{1}{5}} \tag{2.2.19}$$

Rayleigh time is proportional to particle size and hence smaller particles require smaller time steps under the Rayleigh time criterion. On the other hand Hertzian time is inversely proportional to relative velocity which is a system characteristic, hence higher velocities warrant reduced time steps. LIGGGHTS provides functionality which computes Rayleigh and Hertz times at user specified intervals and returns ratios of computed measures and current time step.

Material and Physical Properties

Material and contact properties are required to quantify stiffness and damping constants implemented in collision equations *2.2.13 a* and *2.2.14 a*. One such contact property is the Coefficient of Restitution (COR) a dimensionless measure of collision momentum transfer and absorbed deformation or heat energy, defined by the following relations.

$$e = \frac{v_{B2} - v_{A2}}{v_{A1} - v_{B1}} \tag{2.2.20a}$$

$$e = \sqrt{\frac{h_d}{h_b}} \tag{2.2.20b}$$

A purely elastic collision results in a COR of one and a purely plastic collision results in a COR of zero. A typical experiment to determine the COR is done by dropping a ball of material type A onto a surface of material type B. The COR is proportional to the ratio of initial drop height to the bounce height as in equation *2.2.20 b*.

The Coefficient Of Friction (COF) is required by both Hertzian and Hookean contact models. The COF required is the static one [61] which is typically measured by incrementally inclining a surface of material *one* with a block of material *two* resting on it, until the block begins to slide down the incline. Using accelerometer data the frictional force may be derived and hence the frictional coefficient can be determined. Once the particle slides a kinetic COF comes into effect which is generally less than the static COF, hence the static COF is used as the proportionality constant for Coulombs frictional

limit.

Densities, Youngs Moduli and Poissons Ratios are readily documented quantities, although the values should be verified by testing relevant materials which are likely to form part of the real system. Contact quantities such as coefficients of friction and restitution are subject to higher variability and hence should be tested for between the particular materials comprising the design system. Collision modelling is not of primary interest in this proof of concept study and hence the selected material and contact property values are somewhat arbitrary as long as they do not dictate the system behaviour and compromise results. Steel duct walls and silica quartz particles were assumed in the study.

Silicates have various crystalline forms and hence various material properties associated with each. Many studies have been dedicated to quantifying such material properties. Silica can be found in many forms, some natural and others man made. Steel on the other hand can be produced with various percentages of constituent elements but generally applicable globally averaged values are commonly used. Coefficients of friction and restitution also vary with impact angle and velocity of impact hence these quantities are difficult to specify for computational simulation. In certain cases direct specification of tangential and normal stiffness constants is more suitable for simulation work. Banks *et al* [2] produced coefficient of friction and restitution data for quartz sand particles impacting basaltic rock from wind tunnel testing. The angle of impact was varied by pitching the basaltic rock in the wind tunnel. The quartz particle size was around 500 micron. Friction coefficient data was produced for 60° impact at 11 ms^{-1}. Restitution coefficient data was produced over a range of impact velocities and angles. Rock on rock frictional coefficients were specified within consistent ranges in a paper by Toro *et al* [36] and in a Stanford GEOL 615 document. Sommerfeld *et al* [32] produced coefficient of friction and restitution data for quartz particles impacting untreated steel surfaces comprising a channel geometry at various angles and velocities. The quartz particles were around 100 micron in size. Kim *et al* [18] produced quartz scratch test friction coefficient data. In the book *Particle Breakage* [53], measured and calculated restitution coefficients were documented pertaining to quartz steel impact. It was assumed that the coefficients of friction and restitution were equivalent in reciprocal contact i.e. silica on steel and steel on silica. Bulk Silicon had a higher Youngs Modulus range around 130-190 GPa compared with Silicon Dioxide and Fused Quartz which was documented around 50-70 GPa in work by Drane *et al* [7]. Silicon densities and densities of Silicon derivatives also varied somewhat but could be grouped within a reasonable range, and similarly with Poissons ratios. Refer to appended tables *B.2* and *B.3*.

The tabulated quantities were used in the study pending subsequent revision. The input script may simply be edited with respect to these entries if warranted. In addition to the previous considerations certain coefficients are irrelevant i.e. steel on steel contact coefficients as contact of this nature cannot occur in the system where steel duct walls are stationary, nevertheless these entries must be specified in the input script. The simulation may accommodate for material properties which fall within specific ranges such as density. Size range may also be specified but is not a material property and will be left out of this discussion. Since the particle size range used in the study falls below that for which the tabulated values were derived, the values may be in a certain degree of error due to scaling effects which were unaccounted for. Material properties would not have an effect on the acoustically

driven nodal and anti nodal positions in the assumed idealized system. In reality the reflecting duct wall boundary material would affect acoustic reflection characteristics. Additionally particles may also affect acoustic characteristics by virtue of scatter phenomena which are dependent on particulate phase material properties. Scatter would be more pronounced in flows with high particulate concentrations. The idealized study did not consider the implications of material properties on acoustic phenomena.

Simulation time step would have to be reduced to produce sufficient collision resolution. Considering dilute systems where collisions have no global effect and have low probability of occurrence, the reduced time step and the additional computational time required as a result would not be justified. Contact forces may be avoided by selecting a large timestep to allow particles to cross each other completely without resolving any contact. Alternatively no contact behaviour may be flagged in the input script. Collision dominated systems would require more sensitive material and contact property specification and perhaps subsequent adjustment to compliment validation data.

2.2.3 CFDEM

The main Computational Fluid Dynamics coupled to Discrete Element Method (CFDEM) code is built on the standard incompressible transient PISO algorithm. The momentum exchange between continuous and discrete phases is accounted for in the NSE by introducing a discrete $\mathbf{R}_{sl}$ particle momentum exchange field and a voidfraction α_f which premultiplies all terms in the original NSE. The modified NSE take the form of equation *2.2.21 a* [20, 25]. The $\mathbf{R}_{sl}$ momentum exchange field is further decomposed for numerical reasons, into explicit and implicit terms as in equations *2.2.21 c* and *d* respectively. The basic structure of the predictor corrector algorithm as outlined in equations *2.2.11 a-l* is preserved, whilst accounting for the fluid-particle momentum transfer in the pressure and velocity correction equations. The pressure correction equation *2.2.11 h* has an additional term accounting for the particulate phase which remains after taking the divergence of the flux corrector equation *2.2.11 f* containing the particulate field contribution.

$$\frac{\partial \alpha_f \overline{\mathbf{U}}}{\partial t} + \alpha_f \boldsymbol{\nabla} \cdot \left(\overline{\mathbf{U}} \otimes \overline{\mathbf{U}} \right) = - \left(\alpha_f \boldsymbol{\nabla} \cdot \mathbf{R} - \alpha_f \boldsymbol{\nabla} \cdot 2\nu \overline{\mathbf{D}} \right) - \mathbf{R}_{sl} - \alpha_f \frac{\boldsymbol{\nabla} \overline{p}}{\rho_f} + \alpha_f \mathbf{F} \qquad (2.2.21a)$$

$$\alpha_f \boldsymbol{\nabla} \cdot \mathbf{R} - \alpha_f \boldsymbol{\nabla} \cdot 2\nu \overline{\mathbf{D}} = -\boldsymbol{\nabla}^2 \left(\alpha_f \nu_e \overline{\mathbf{U}} \right) - \boldsymbol{\nabla} \cdot \left[\alpha_f \nu_e \, \mathbf{dev} \left(\boldsymbol{\nabla} \overline{\mathbf{U}}^T \right) \right] \qquad (2.2.21b)$$

$$\mathbf{R}_{sl} = \frac{\mathbf{K}_{sl} (\mathbf{U_f} - \mathbf{U_p})}{\rho_f} \qquad (2.2.21c)$$

$$\mathbf{K}_{sl} = \frac{\mathbf{F}_i}{|\mathbf{U_f} - \mathbf{U_p}| \, V_{cell}} \qquad \mathbf{F}_i = \sum_{f=1}^{N} \mathbf{F}_f \qquad (2.2.21d)$$

The CFD-DEM coupling is achieved by consecutive alternating CFD DEM runs. LIGGGHTS accounts for all fluid induced particle forces as well as those forces external to the fluid particle interaction environment. The CFDEM code identifies CFD volume cells which contain the respective particles and computes the volume fractions occupied by the discrete phase within the containing cells. Relative fluid particle velocity components are also computed. Particle containing cells have to be identified so

that discrete momentum exchange fields may be formulated according to the underlying Volume Of Fluid (VOF) mesh. The momentum exchange terms are then computed and fed into the NSE which are then iteratively solved according to equations implemented in the modified PISO algorithm. The fluid particle forces are computed and sent to the DEM solver which then increments particle velocities and positions accordingly and the process repeats until a user specified end time has been simulated [25].

Two models namely *'A'* and *'B'* can be selected to either include or exclude respectively fluid solid void fraction in the modified NSE equation *2.2.21 a* which is documented according to model *'A'*. Dilute systems or systems with small particles contribute negligibly toward void fraction hence the inclusion thereof would have minimal effect. Relevant force models are selected by the user. Resulting forces acting on particles are summed to produce the implicit force vector denoted by $\mathbf{F}_i$ which is then carried through into the momentum equation via the $\mathbf{R}_{sl}$ field. The $\mathbf{R}_{sl}$ field is effectively the equal and opposite continuum force which the particles exert on the fluid. All CFDEM specific fields including vector fields $\mathbf{R}_{sl}$ and $\mathbf{K}_{sl}$ and scalar field α_f are continuum discrete by virtue of their dependence on the discrete particulate phase.

Generally the most definitive force acting on suspended particles is drag [20, 25]. The numerous additional forces detailed in section *2.2.2* may be comparable in magnitude to drag force in certain regions of flow or under particular flow characteristics. Different forces have orders of magnitude relevance and their applicability to certain flow systems can be determined before hand based on qualitative preliminary system evaluation.

The current study was limited to including only Di Felice drag and Archimedes buoyancy forces. Since the particles in simulation would be introduced into bulk flow away from any walls and wall velocity gradients, Saffman shearing and Magnus rotational forces would be negligible. In addition due to the steady nature of the flow profile and the inability of the RANS eddy viscosity model to resolve vorticular and secondary flow, local fluid and particle accelerations can be neglected hence effectively eliminating Basset and virtual mass force contributions. The pressure gradient force may be applicable but considering the proposed size of particles in suspension, between 10 and 60 micron, the force may be orders of magnitude smaller than the drag force. The omitted forces can easily be introduced, with the exception of the Basset force which is not supported, without compromising the original system. Negligible forces would remain negligible even if they are included in simulation. The only cost incurred would be computational time. If particles were expected to migrate to the walls then shearing and spin related forces would be applicable, however since RANS does not produce cross currents particles cannot be eddied toward the walls from the bulk.

The Di Felice drag model equation *2.2.22 b*, based on an earlier formulation by Gidaspow was implemented in the study, in preference over the Schiller Neumann model which is more readily found in literature. Di Felice drag had been used successfully in papers by Kloss and Goniva *et al* [20, 25] in which promising validation simulations were outlined. In addition Di Felice drag has been recommended by Kloss and Goniva as a first pass model applicable to both dilute and dense regimes without discontinuity as was the issue with the previous Gidaspow model [20]. Equation *2.2.22 b* and its compliments reflect that force implemented in the code. The gravitational contribution is omitted

in the Archimedes model of equation *2.2.22 a* as it is not a fluid induced force and is dealt with separately in LIGGGHTS.

$$\mathbf{F_1} = -\mathbf{g}\,\rho_f\,\frac{\pi d_p^3}{6} \tag{2.2.22a}$$

$$\mathbf{F_2} = \frac{1}{2}\rho_f\,|\mathbf{U_r}|\,\mathbf{U_r}\,C_D\frac{\pi d_p^2}{4}\,\alpha_f^{2-\chi} \tag{2.2.22b}$$

$$C_D = \left(0.63 + \frac{4.8}{\sqrt{Re_p}}\right)^2 \qquad Re_p = \frac{\alpha_f|\mathbf{U}_r|d_p}{\nu}$$

$$\chi = 3.7 - 0.65\,\exp\left[-\frac{(1.5 - \log_{10} Re_p)^2}{2}\right]$$

2.3 Previous Work

To the authors knowledge at the time of writing, not much literature was available on the subject in general and very little on particle agglomeration in large scale chambers apart from experimental work. Almost no computational work was available on large scale deflection application, therefore a contextually broad review is presented to give the reader an idea of what has been done. Research topics are listed below and some are detailed further:

- **i** Industrial scale acoustic system chamber-transducer geometry optimization [16], overview of new industrial scale applications utilizing acoustic fields [8] and an experimental testing rig to determine industrial efficiency parameters [4]
- **ii** Modelling of acoustically driven pressure field non-linearities [5, 39, 17, 38]
- **iii** Various experimental rigs of slightly different design and operational principles accompanied by numerical validation models [15, 10, 37, 13, 12]
- **iv** Contributors to the theoretical description of the phenomena King, Yosioko, Kawasima, Gorkov, Doninikov, etc
- **v** Experimental stationary fluid rigs designed for testing under various gravitational fields [11, 1]
- **vi** Theoretical analysis on the possibility of stirring and mixing fluids within micro cavities using acoustic forces [30]
- **vii** Theoretical analysis of Gorkov's particle interaction potential fields in cylindrical, spherical and plane standing wave geometries [3], experimental rig to validate Gorkov field theory in a cylindrical waveguide [41]
- **viii** Layered resonator modelling for micro-chamber application [14]
- **ix** Numerical modelling of bubble-bubble and bubble- acoustically induced pressure field interaction [51] and bubble-bubble extended Bjerknes theory interaction [6]
- **x** Experimental investigation of microstreaming phenomena in micro cavities [34, 33]

The current study is geared toward large scale modelling hence papers [4, 16, 5] are more relevant

than papers dealing with micro chambers and effects which predominate small scale applications such as microstreaming and thermally induced convection, however these can provide technical insight.

2.3.1 Macrochamber Agglomeration Studies

In work presented by de Sarabia [4] an experimental rig was designed which used an acoustic forcing pre-filter, referred to as an acoustic agglomerator, to achieve agglomeration followed by an electrostatic precipitator for use in particle removal from coal combustion fumes. Agglomeration is induced through orthokinetic i.e. capture volume by virtue of particle wakes, and hydrodynamic i.e. collision interaction, means. All numerical studies were validated through experiment and found within good agreement validating the orthokinetic component of agglomeration. The study mentioned particle wake effects as being responsible for attractive particle forces. The study utilized a driving frequency range between 10 and 20 kHz on micron and submicron sized particles.

Gallego-Juarez [16] produced a paper in which transducer plate geometry was optimized for use in industrial scale applications. The need for high power transducers with large radiating areas was highlighted and achieved by introducing a mechanical vibration amplifier in series with the PZT. The paper also mentions the importance of impedance matching between the transducer and transport fluid or gas (typically highly absorptive) resulting in reduced acoustic wave intensity away from its source. Rectangular stepped plates were found to produce best results with respect to directivity and uniformity of vibrational distribution. A previous circular transducer design (also stepped) could produce maximum intensity levels of 165 decibel (dB) between 10 and 40 kHz at 1 kW power. Finite Element Method (FEM) was used for transducer design and Boundary Element Method (BEM) for the resulting acoustic field. A prototype was designed sizing in at $1.8 \times 0.9\ m^2$. A laser vibrometer was used to measure radiating power which was compared to input power to produce an efficiency of 67% at a power capacity of 5 kW. A cubic chamber was used to test the transducer with an adjustable reflecting wall to fine tune the resonance in the chamber.

Earlier work by Gallego-Juarez *et al* 1994 [8] resulted in a overview document highlighting the usefulness of the technology and promising areas of its application. The previous design optimization detailed in [16] was in fact built upon the design documented in 1994. The circular stepped plate was replaced by a square stepped plate to improve the uniformity of pressure distribution. The stepped design was a marked improvement over the regular flat plate actuator which produced poor directivity. The new model was promising due to its high power capacity, large pressure amplitude, efficiency in converting electrical energy to mechanical vibration and uniformity of distribution across its vibrating interface. Power capacity is critical, measured by the amplitude of vibration which the design should tend to maximize. By adjusting the sizes of the actuator surface and steps any acoustic field configuration could be obtained. The paper also mentions the use of Finite Element Modelling to optimize and correct design geometry. The circular actuator was tested in particle removal from emissions, defoaming and cleaning of textiles. Agglomerators seem to have had the best success from a commercially viable point of view. The early circular model had been applied to a 2 m long 220 mm diameter drum at 150 kW and 160 dB to agglomerate 0.5 micron average diameter, 2.2 micron deviation carbon black smoke particles. A two to three second treatment period had been found effective. Another precondi-

tioner unit operating on the same principle was installed upstream of an electrostatic precipitator in a coal fired utility boiler system where submicron particulates are common. The unit consisted of four actuators vibrating into a horizontally orientated rectangular sectioned chamber operating between 10 and 20 kHz at pressures between 140 and 170 dB at 400 W each. A cooling system was also introduced to protect the actuators from overheating. The preconditioner unit produced larger agglomerates and in so doing increased the efficiency of the electrofilter by 5%. The defoaming phenomenon has been attributed to a combination of mechanisms including large acoustic pressures, radiation pressure, bubble resonance, streaming etc. The same circular resonator had been implemented successfully in a high speed canning foam control system where the acoustic field was beamed onto foaming surfaces. Textile cleaning has also been a potential area and industrial prototypes have been built but the technology has not found as much success and will not be elaborated on.

2.3.2 Microchamber Deflection Studies

The papers which describe various experimental rigs will be outlined in less detail as they are all based on similar principles of laminar flow and linear acoustic theory. All models reviewed include individual particle tracking by considering all relevant forces acting on any given particle. Secondary forces are mostly neglected and only major forces including primary acoustic, gravity, buoyancy and drag are considered. Particle interaction is avoided in all reviewed papers dealing with particle trajectories. Johnson and Feke [15] produce a model in which two streams are fed into the acoustic chamber from the left and right and flow vertically, mix within the chamber and are then split again downstream of the acoustic field. The right stream contains sediment and the left stream does not. In the acoustically driven region the particulates are driven from the contaminated stream into the path of the extractor stream. This is known as acoustic washing. The downstream splitter plate serves to divert the now contaminated stream left and allow the newly clarified stream to exit right. Their numerical model made use of CFD to produce the flow profiles slightly deviated from the regular laminar profile due to the presence of splitter plates. The CFD flow profile was then used to compute drag forces on the particles which were subsequently fed into a numerical model applying additional forces, and used to ultimately calculate required residence time, length of transducer required, etc. Their model was verified experimentally. Secondary acoustic forces were neglected under the assumption of their lower order of magnitude compared with primary acoustic forces in dilute suspensions. A similar splitting concept was presented by Gupta *et al* [10] making use of the acoustic contrast factor to split particles of different compressibility into preallocated streams. Polymer recycling systems typically contain particles of similar density and size but with different compressibility by virtue of their moduli. The proposed system splits particles according to their respective acoustic contrast factors followed by diverting streams or flow splitters to extract them either from the peripheries of the chamber or from the central axis of the stream. A simplified trajectory model was implemented to compute the heights of the flow splitter plates required in conjunction with the length of the acoustically driven deflection region. Laminar flow was assumed along with a simplified particle force model.

Two papers by Hawkes and Coakley [12, 13] deal with filtration systems which they claim could be scaled either way for implementation in macro or micro systems. Again a laminar profile was assumed

and developed through an entrance length. The assumption or realization of laminar flow ensures that the converged particulate concentration within the respective pressure plane does not disperse once past the acoustic field, an important consideration in positioning the downstream splitter plate. Their 2001 paper [13] notes the implementation of a $10 \times 0.25\ mm^2$ flow cross section. Chamber wall thickness was an important design consideration and should be dimensioned with respect to the acoustic wavelength within that material in the acoustic section. The transducer was driven by an amplified sinusoidal signal and chamber resonance was detected at voltage maxima. A number of resonant modes could be detected at subsequent voltage peaks and these were investigated individually to determine which produced best filtration efficiency. The earlier 1996 paper described yeast filtration at diameter size 5 micron, whilst the more recent 2004 paper focused on polystyrene latex contaminant filtration at diameter size ranges between 1.5 and 25 micron and both report filtration efficiencies greater than 90%. Reduced efficiencies could have resulted from cavitation or thermal gradients which cause diffusion, convection and have been attributed to disruption of the standing wave pattern [12]. Cavitation was however not observed and temperature increases were low but nevertheless attributed to a reduction in filtering efficiency. The system designed in [13] was reported to have high levels of stability i.e. the system was tested over a 6 month period with no variation in maximum efficiency. Their earlier 1996 design [12] was based on agglomeration and sedimentation mechanisms to achieve yeast filtration. A critical sedimentation radius of 49 micron was determined comprising 9669 yeast cells with an average diameter of 4.6 micron assuming a normally distributed size range. Highest efficiencies were reported when flow rate was adjusted in accordance with particulate concentration. Smaller particulates were linked to reduced efficiencies. The 2004 paper notes that using multiple copies rather than scaling geometric dimensions was more suitable for incorporating their design into larger systems.

2.3.3 Particle Manipulation

Work by Yutaka *et al* [1] produced interesting results which contradicted theoretical knowledge describing bubble behaviour in acoustic fields. A vertically orientated experimental rig was designed which allowed bubbles to be injected at the bottom end and rise up through the vertical chamber under buoyant drive. Bubbles sized 2 to 3 mm were produced. The acoustic field would then be applied within the chamber along its vertical axis in the hope of trapping the bubbles against buoyancy. The transducer was driven by a function generator as in [13] and the vibrational displacement was amplified through a horn. Theoretical models predicted that bubbles would be forced toward anti-nodal planes in most cases, however the experiment showed that bubbles were trapped between nodal and anti-nodal planes. Smaller bubbles were captured whilst larger bubbles escaped. Mid sized bubbles oscillated about equilibrium positions before settling. The accompanying numerical model accounted for change in bubble radius, via the Rayleigh-Plesset equation, brought about by external spatial pressure variation as the bubble traversed the pressure field. The numerical results agreed with those obtained experimentally with respect to which bubbles would be trapped and which would escape, as well as predicting the equilibrium position between node and anti-node, the offset which was attributed to the buoyancy force. Additional experimentation was done by introducing two vertical actuator horns facing one another through the fluid medium. The phase difference between the

transducers was varied accordingly to effectively shift the standing wave field, and thereby moving the bubble equilibrium co-ordinate and hence the bubble itself. This experiment was repeated under parabolic flight conditions to produce gravitational variation. A bubble moved to the nodal position in reduced gravity attributed to reduced buoyancy. Under hyper-gravity the bubble was retained by the acoustic pressure field. The bubble also changed shape under hyper-gravity. An 8 *mm* diameter bubble was trapped using the apparatus in reduced gravity which was not achievable under normal gravity. Vertical displacements of 5 *mm* were achieved by applying frequency shifting.

The work of Barmatz *et al* 1984, [3] produced mathematically derived point locations or region potentials of influence, based on Gorkov's theory, which would draw particles, within various geometries by activating various acoustic modes i.e. acoustic resonant propagations in three dimensions and the resulting interference field. The work investigated the possibility of precisely positioning a particle within a resonant cavity by activating various resonant modes for manipulation or examination purposes. An earlier paper by Whitworth and Coakley 1991 [41] examined the effect of cylindrical chamber diameter on particulate alignment at half wavelength intervals along the height of the cylinder which related to the work published by Barmatz *et al* [3] where resonant modes in cylindrical cavities are described. The paper by Whitworth noted that particles moved outward toward the waveguide extremity within their respective nodal planes when the waveguide dimension was narrow and moved inward forming striated axial columns when the waveguide dimension was increased. This same phenomena was also noted by Hawkes *et al* [11]. These migrations correspond to the theoretical potential field based on Gorkov's theory which manifests within the geometric cylinder under specific modal excitations. The early work of Barmatz had only accounted for primary acoustic forces in particle positioning application. Various particle interaction regions of potential apart from isolated points were show to develop including lines, circles, spheres, planes and cylinders. It was also found that for each geometry considered i.e. spherical, cylindrical and planar, a set of modes could be excited to produce an isolated potential in the centre of the cavity. These potentials were verified experimentally for the case of a rigid sphere. The acoustic effect on bubbles in liquids was also noted as an area of potential interest. The paper mentions the possibility of incorporating additional fields into Gorkov's acoustic potential field, resulting in a complete force field accounting for a number of different forces including gravitational, acoustic, etc.

2.3.4 Computational Studies

Kaltenbacher *et al* 2002 [17] describes computational models including various coupling schemes to simulate piezoelectric and electromagnetic actuators. A Piezoelectric-Acoustic coupling model was designed as well as an Electromagnetic-Mechanical-Acoustic coupling one. Their computational design allowed simulation of non-linearities from both transducer and wave field, as well as linear simulation for comparative purposes. Three test cases were investigated modelling a high powered pulsed piezoelectric source, an electromagnetic acoustic power source and a high intensity continuous standing wave generator. The pulsed source simulation revealed a 33.7% discrepancy in peak pressure due to non-linear effects i.e. the non-linear pressure peak was 33.7% higher than that simulated based on linear theory. The electromagnetic pulse source simulation was designed to mimic human kidneys

and urinary tract. The simulation results were in agreement with experimental data. Standing wave simulations had additional non-linear effects. The piezoelectrically driven standing wave transient simulation test case revealed that the system energy increases until non-linearities come into effect and that part of the energy is transferred to higher harmonics. These types of simulations provide further detailed insight which could help optimize industrial scale systems and reveal possible application potentials.

Other computational studies, implementing finite element, finite difference and finite volume schemes, including those done by Doehle *et al* [5] and Vanhille *et al* [39], deal with numerical modelling of Kuznetzov's set of non-linear equations in cylindrical resonant tubes. Cylindrical chambers are commonly simulated because there is sufficient analytic and experimental data available for validation. The models developed were specifically geared toward industrial scale acoustically driven separators and agglomerators, highlighting the importance of pressure, the variable field of interest, in achieving appreciable acoustic forces. Both papers showed simulation results to be in agreement with experimental data and analytic forms.

CHAPTER 3

METHODOLOGY

3.1 Model Development

This section will outline the study by introducing the computational model in a general sense touching upon the geometry of the control volume and boundary conditions. The assumptions and limitations of the model will also be addressed alongside brief overviews of the software used. A program flow diagram will be presented giving an overview of the processes involved in obtaining a final solution.

3.1.1 Geometry

The vertically orientated parallelepiped control volume depicted in schematic figure *3.1* used in this study had a square base cross section of length 50 cm and a height of 70 cm in the flow direction. The geometry was defined in Cartesian three dimensional space with flow parallel to the positive z direction. The flow was driven by a pressure gradient which was adjusted between inlet and outlet accordingly to achieve various flow profiles distinguishable by maximum flow velocity at the centreline of the duct. A number of particles would be randomly inserted in a small region about 10 cm from the inlet as marked in the schematic. The particles would be subject to fluid drag force acting in the z direction and an imposed acoustic force acting in the x direction deflecting them to various nodal planes along the cross section of the duct. The acoustic force was assumed active along the entire height of the control volume as depicted in the schematic by the transducer.

Six simulations were run in total, three purely based on developing flows of different centreline velocities, the remaining three coupled with those previously developed but focusing on particulate response. The three flow velocities attained at the duct centreline by adjusting the pressure gradient across the height of the control volume were 0.1, 0.5 and 1.0 ms^{-1} respectively. The fluid was modelled as water, density 1000 kgm^{-3} at 20° C. For each converged flow profile two driving frequencies were tested, one below the ultrasonic marker of 20 kHz at 14794 Hz, and one above the marker at 26629 Hz. The driving amplitude was kept constant at 50 micron. A uniform particle radius distribution in the range 5 to 30 micron was assigned to the 20 particle insertion, along with a uniform density distribution range between 2300 and 2700 kgm^{-3}. The density distribution was selected based on quartz data encompassing a variety of crystalline forms found in nature. A size distribution was assigned so that the effect of radius on the magnitude of acoustic force could be isolated. Flow velocities were selected

to cover an appreciable range.

The flow simulation was expected to remain invariant in the coupled framework due to the fact that a converged solution is mapped across and because of the short simulation time required for the particles to traverse the simulation domain. As a result all flow related data and plotting is generated during the flow development run, and all particulate data is recovered from the coupled simulation. The coupled simulation time was chosen based on the time it would take the particles to traverse the height of the control volume. Higher flow rates would require shorter simulation times as the particles would be carried faster through the simulation domain by virtue of larger drag forces.

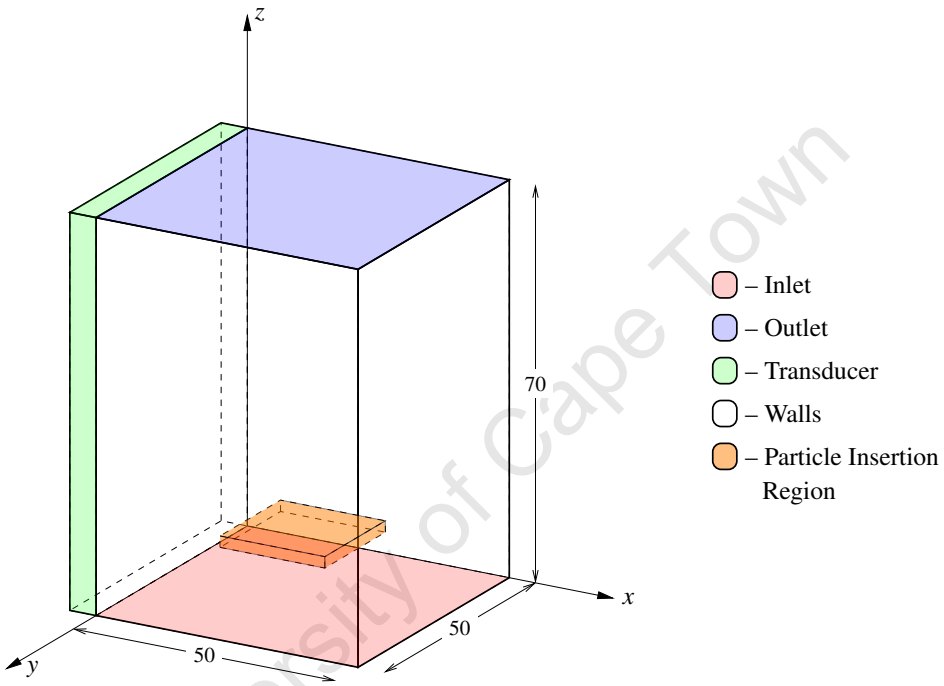


Fig. (3.1) *System schematic*

3.1.2 Assumptions and Limitations

Simplifying Assumptions

A number of simplifying assumptions had been made throughout the development of the model. Simplifying assumptions introduce various degrees of error in the system. Real models are built up from basic ones, systematically eliminating assumptions, and thereby increasing complexity and simulation accuracy. Physically representative models require more accurate solutions whereas proof of concept models do not require high levels of accuracy as was the case in this study. The assumptions made are listed

i The carrier fluid is incompressible and uncontaminated

ii Turbulence is isotropic modelled using an eddy viscosity $k-\epsilon$ model. Eddy viscosity models are unable to account for secondary flows

iii Thermally isolated and isothermal system i.e. no heat exchange from the boundaries and no internal heat generation

iv Particles are spherical and do not interact with one another. This is achieved by implementing a large time increment which is essentially inadequate to resolve collision

v Particles remain within the bulk flow and do not approach nor interact with wall boundaries. This is partly due to the absence of secondary flows and small scale vortices which are otherwise responsible for eddying particles in planes perpendicular to the primary flow direction

vi Particles do not scatter the acoustic field or do so without affecting its general form

vii The acoustic system is ideal i.e. no energy absorption by the fluid and no losses through transmission at the reflecting boundary. In addition the field is planar with no transverse non-uniformity

Model Limitations

Certain assumptions are plausible for instance **i** or **iii** and do not need to be reconsidered if the system is designed accordingly, whilst other assumptions such as those described in **v**,**vi** and **vii** are made to reduce modelling complexity at a loss of some physical character which is not accounted for adequately or not accounted for at all. The impact of a particular physical character should be assessed to ascertain whether the model needs to be revised and upgraded. System conditions had been tailored to accommodate the use of certain equation forms including the planar acoustic force which is valid only under regular geometry in which a planar wave can propagate without disruption. Limitations are mostly incurred from the set of assumptions made. Some of the limitations are listed

i Turbulent eddies and secondary flows were unresolved hence diffusive particle behaviour remained unaccounted for. Dispersive eddies were expected to interfere with the acoustic system by carrying particles out of nodal planes and generally toward preferential regions of the flow profile. In addition turbulent eddies may produce acoustic scatter which would further affect the efficiency of acoustic deflection. The absence of eddies allows particles to be inserted in isolation of the wall boundary, and remain isolated throughout the simulation run. Wall effects were not accounted for however may become relevant if particles are eddied to the walls. The current model does not have specific treatment for particles in the near wall region

ii Acoustic non-linearity was not accounted for in the model under the assumptions of lossless propagation through the fluid medium and ideal reflection at the opposite wall boundary. Neglecting non-linearity can greatly overestimate the magnitude of the acoustic force and pressure form away from the transducer source. The error is expected to grow as one samples further from the actuating surface.

iii Particles may themselves scatter acoustic waves, the effect which has been noted in literature would be more notable in dense systems however dilute systems have also been found to exhibit acoustic scattering phenomena. The degree of scatter would be dependent on a number of acoustic parameters as well as particle material and fluid properties. It should also be noted that contaminants can affect the acoustic form through similar mechanisms

iv The primary acoustic force equation discussed in section *2.1.3* is only applicable to planar acoustic standing wave fields. A more complex root level form of the equation incorporating spatial pressure field gradients would have to be used when modelling irregular geometries. Closed analytic forms exist for other simple geometries

v Particle contact is not resolved under the time step adopted in the study, however dilute system particulate phase collisions were not expected to interfere with the function of the acoustic force. However if agglomeration phenomena is of interest additional particles would have to be introduced, cohesion properties defined, additional short and longer range force models included and time step reduced. This is possible under the current model at a cost of additional computational time, and hence is not a model limiting factor but rather one of time. Any collision dominated system is time intensive because of the small time step required to resolve it. Stiffness and damping coefficients are determined from material and contact properties supplied by the user, subject to considerable variability error

vi The current model has to be upgraded to account for other types of particles such as gas bubbles and liquid droplets each with their own set of accompanying assumptions

3.1.3 Modelling Tools

OpenFOAM

The development of OpenFOAM began in the late 1980's at the Imperial College of London in the hope of producing a continuum mechanics computer library. A previous version of OpenFOAM called FOAM was sold into the commercial market, however it re-emerged in 2004 as open source software under the name OpenFOAM. The code was made open source to assist a large user base of CFD researchers, developers and consultants. The primary developers of OpenFOAM include H. Weller, G. Tabor, H. Jasak and C. Fureby [40]. The first public domain version 1.0 was released in December 2004 and has been under continuous development since. OpenFOAM has been tested on Windows, Macintosh and Unix operating systems.

OpenFOAM is a continuum mechanics code which provides for parallel computing limited to 1000 Central Processing Unit (CPU)'s, by implementing domain decomposition and reconstruction. It is more well known as a CFD code but is geared toward any continuum mechanics problem. OpenFOAM is based on finite volume polyhedral discretization i.e. regular cell elements are required however efforts are under way to accommodate irregular meshing. One of the great strengths of OpenFOAM is its tensorial, differential equation mimicking which allows as far as possible the use of natural continuum mechanics notation in the code. OpenFOAM has been used to model compressible and incompressible flows, Euler-Euler multiphase flows, Lagrangian particle tracking, RANS and LES incompressible and

compressible turbulence flows, Direct Numerical Simulation (DNS), free surface flows, thermophysical models as well as potential flows, fluid structure stress analysis and even magnetohydrodynamics [43, 42]. OpenFOAM is essentially a continuum mechanics library written in C++ and developed on Linux, which can be easily extended at higher levels due to its object orientated structure. As with any CFD code a three step method is adopted, preprocessing, solution and post-processing. The code is supplemented with many utilities and solvers which are useful for data extraction, mesh generation and manipulation, analysis and pre and post-processing. OpenFOAM can produce output in a number of different formats for compatibility with external visualization applications. Simulation studies have shown that OpenFOAM compares well with commercial packages such as FLUENT and has a good track record with respect to non-linear and strongly coupled problems.

Gmsh

"G Mesher (Gmsh) is an open-source three-dimensional finite element grid generator with a build-in Computer Aided Design (CAD) engine and post-processor" [9]. For the purposes of this study Gmsh is an open source one, two and three dimensional meshing tool written in C++, which can be run on laptop computers and clusters alike. The project was initiated in 1997 by Christophe Geuzaine and Jean-François Remacle and has been made compatible with Linux, Macintosh and Windows operating systems over the years. The developers outline three key aspects of Gmsh namely fast, light and user friendly. Fast in terms of meshing time and Graphical User Interface (GUI) interaction response time, light in terms of memory and third party compliments and user friendly in terms of code extension for developing purposes and an 'easy to use' interface.

Gmsh comprises four main modules namely geometry, meshing, solving and post-processing [46]. The geometry module can be used as a CAD interface. The mesh module can be used to produce structured and unstructured grids, structured grids being suitable for integration into OpenFOAM, and unstructured surface triangulated meshes in Standard Tessellation Language (STL) format for use in LIGGGHTS as complex wall boundaries. Numerous meshing algorithms can be selected from to compute the mesh. Gmsh has been used effectively in generating repetitive geometry through programming language functionality such as looping and parametrization to allow simple model and component scaling. It also allows for the generation of self containing input scripts containing variable assignments and computations, as well as providing for specific control of mesh element size and shape. Gmsh is efficient, can be run on low end machines and compiled without the GUI.

LAMMPS and LIGGGHTS

The following information can be found in the LAMMPS manual [54] or alternately as part of the LIGGGHTS online documentation [61] and on the LIGGGHTS website www.liggghts.com. The core paper dealing with fast parallel molecular dynamics algorithms, which lead to the development of the LAMMPS code was published in 1995 based on a technical report produced in 1993 by Plimpton for Sandia National Laboratories [27]. LIGGGHTS is a more recent DEM code based on the previous molecular dynamics code LAMMPS. LIGGGHTS is an extension of LAMMPS with respect to

granular dynamics capabilities including heat transfer and is aimed toward industrial system simulation. LIGGGHTS was based on a paper detailing CFD and coupled CFD-DEM models, algorithms and validation, authored by numerous members of the Christian Doppler Laboratory on Particulate Flow Modelling (CDLPFM) at the Johannes Kepler University (JKU) Linz, Austria [21]. The core developer of LIGGGHTS was C. Kloss.

LAMMPS is a classic molecular dynamics C++ code which can be used to model atomic, polymeric, metallic, biological, granular and coarse grained systems in solid, liquid or gaseous states using a variety of force fields and boundary conditions. Single particle systems as well as systems with millions and billions of particles have been modelled. LIGGGHTS has been used to simulate one million particle systems on desktop computers [20]. The previous LAMMPS granular models have been redefined including new wall collision and cohesion theories. In addition to these improvements CAD geometry imports including stress analysis have been introduced as well as heat conduction models for spheres and moving meshes [25]. LAMMPS can run on single processor units and on parallel multiprocessor units through a Message Passing Interface (MPI). Additional functionality through high level user defined codes is easily integrated into the object orientated environment supported by a well designed hierarchical code structure. Different inter-particle force field models are available to choose from including Derjaguin, Landau, Verwey and Overbeek (DLVO), colloidal, long and short range Coulombics, Lennard-Jones along with many others. LAMMPS can deal with two or three dimensional problems with numerous boundary wall conditions and rigid body inclusions. Various output formats are supported for different post-processing applications.

LIGGGHTS has been used to simulate numerous physical systems, including granular flows with complex walls such as in hoppers, moving or rotating walls such as in rotary driers as well as moving buckets, non-spherical particles by clumping spherical ones, flat or curved conveyor transport, wear prediction through abrasion theory, cohesion, granular heat transfer, wave propagation through granular media and chemically reacting systems resulting in changing particle sizes. Future releases will include 6 Degrees Of Freedom (DOF) rigid body dynamics where solid imported CAD surface geometries can move and collide with particles as well as bonded models which can account for solid elastic and plastic deformation as well as internal stresses. Most of the computation time is devoted to detecting and calculating pairwise interaction forces [25]. Many reputable external authorities including Procter and Gamble United States of America (USA), Intrame Spain, MECALYSIS (Pty) South Africa and Baden Aniline and Soda Factory (BASF) Germany are involved in LIGGGHTS benchmarking studies.

CFDEM Coupling

A CFD-DEM coupling code known as CFDEM has been designed to couple LIGGGHTS DEM and OpenFOAM CFD. The code is however limited to transient incompressible flows. The core developer was C. Goniva, also a member at the CDLPFM, Linz. The need for coupling is clear as most physical granular systems are not dry and involve a carrier or background phase, be it air, water or any other fluid or gas [25]. Coupling commercial codes has the disadvantage of limited licence with respect to the number of parallel computers which can be utilized. This drawback is avoided by coupling open source

packages. CFDEM is based on the PISO algorithm and OpenFOAM is utilized as the parent code which spawns intermittent LIGGGHTS DEM runs supplemented by momentum data exchange via MPI. Additional force models can be easily introduced through high level coding. Code extendibility follows the same procedures outlined for OpenFOAM.

The CFDEM code has complete four way coupling capability and has been used to simulate fluidized beds, pneumatic conveying, ball mills, flotation using VOF multiphase gas liquid flow and DEM particles, immersed boundaries, paddle mixers, surge conveyors and hopper discharge. Currently only the CDLPFM is involved in benchmark studies using CFDEM software.

3.1.4 Program Architecture

Each case study comprised of two main simulation procedures, one used to develop a converged turbulent flow profile and map the result into a coupled framework. The second step calling the coupling solver CFDEM governing the transfer of information between OpenFOAM and LIGGGHTS, running intermittently and advancing both fluid and particulate solutions. The CFD solution mapped from the initial run procedure was expected to remain unmodulated in the presence of the dilute particulate phase. The flow was developed using an external shell script **cfdRun.sh**, process diagram *3.2*. The coupled simulation was also catalysed through an external shell script **cfdemRun.sh**, process digram *3.3*. Detailed code dissection can be found in appendix *C*.

The software versions used in this study are tabulated in *3.3*. First releases have been documented previously. The Operating System used in this study was Linux Ubuntu 10.04 Long Term Support (LTS) with two Intel Core2 Duo processors and 8 GiB of memory, each processor at 2.66 GHz. The tabulated version data was valid at the time of writing and is subject to change with time.

The file structure of the coupled case is appended in *A*, figures *A.1*, *A.2*, *A.3* and *A.4*. The directory trees are sufficiently commented to better orientate the reader. Other supplementary files which are spawned from main files during a simulation run have been omitted so as not to obscure the main structure form. All plotters, executables and solvers are called from main run scripts. Three run shells exist namely the **cfdRun.sh**, **demRun.sh** and **cfdemRun.sh**. The DEM related script is run as a preliminary to test the integrity of the input DEM script and will not be detailed further. The CFD related script is run to produce a converged flow profile which is mapped into the coupled case directory. Once mapped the coupled simulation, which introduces the discrete particulate phase into the steady flow, may be run. The **polyMesh** folder contains mesh related files including **points**, **faces**, **boundary**, etc which are not useful at the user level and are hence omitted in this discussion. Additional field variables were required for the coupled case, these being particle related continuum fields **Us** (particle velocity), **Ksl** (fluid particle momentum exchange), **rho** (fluid density) and **voidfraction** (normalized fluid particle volume fraction).

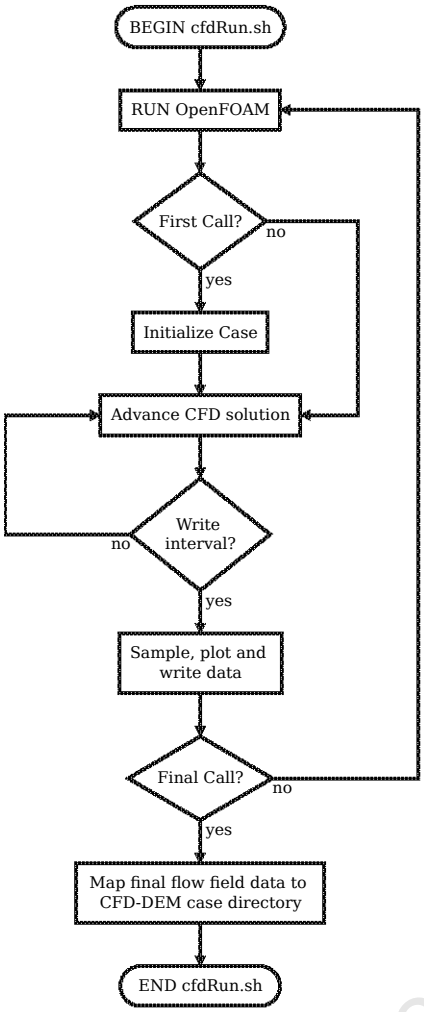


Fig. (3.2) Flow development

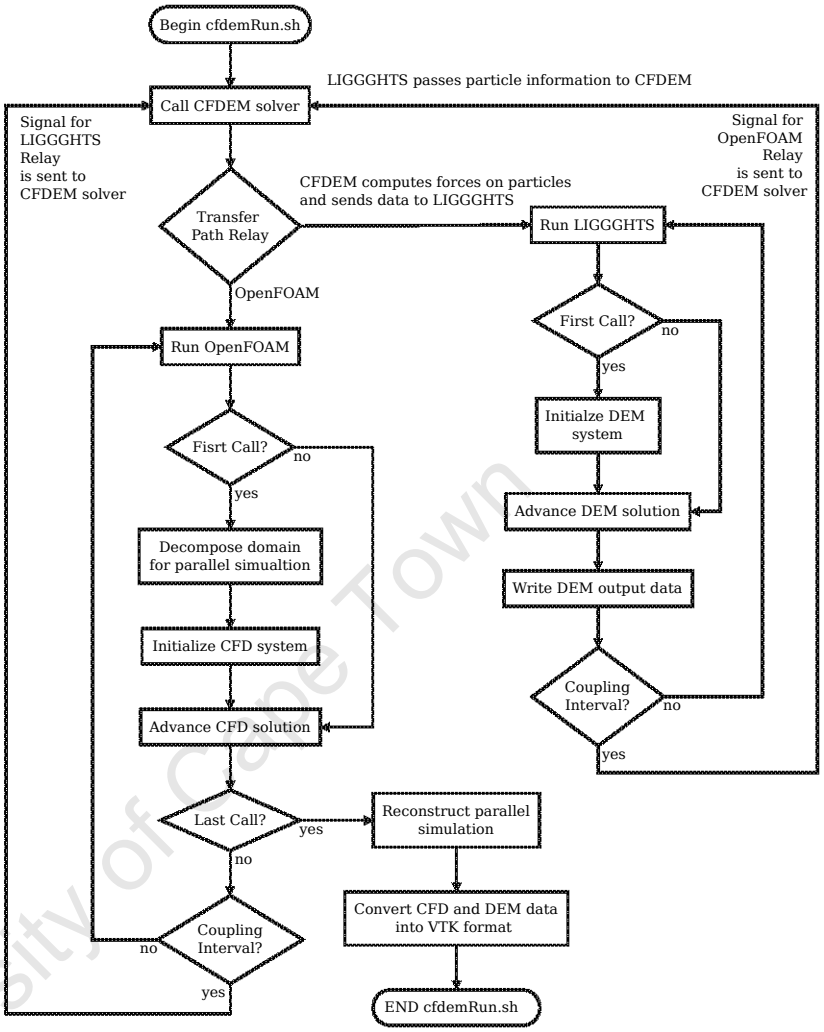


Fig. (3.3) Coupled simulation

Tbl. (3.3) *Software*

software	version	date of release	latest version	date of release
Linux OS	10.04 LTS	Apr 2010	12.04 LTS	Apr 2012
OpenFOAM	1.7.1	Aug 2010	2.1.1	May 2012
Gmsh	2.4.2	Sept 2009	2.6.1	Jul 2012
LIGGGHTS	1.5.0	Nov 2011	2.06	Aug 2012
CFDEM	2.3.0	Dec 2011	2.4.4	Aug 2012
ParaView	3.10.0	Mar 2011	3.14.1	Apr 2012

Time folders which are produced during either pure CFD or coupled simulation runs have been omitted in case directory structures. All sampling related directories have likewise been omitted. Sampling is generally carried out at points, along lines and across surfaces, on which a number of different variables may be sampled simultaneously. Solution convergence data such as residuals are written to a **logs** folder at specified write intervals. All plotting operations are carried out in the **logs** folder. Additional files are generated and passed around throughout the simulation run. Files which are generated include the y^+ y^* file which is updated at every write interval to account for changing y^+ and y^* values. This file is useful when comparing values generated by user defined functions and in-house codes.

DEM output data was produced at specified write intervals which were consistent with those specified for the CFD computation so that time congruent CFD and DEM snapshots could be viewed simultaneously in Parallel Visualization Application (ParaView). The DEM output data was produced in the **post** folder and subsequently converted to Visualization Tool kit (VTK) format through Pizza, a post-processing and data conversion code complimentary to LIGGGHTS. An additional user defined output file is produced in the same folder containing all particle trajectory, velocity, force and time data in matrix form for post-processing in Matrix Laboratory (MatLab).

3.2 CFD Model

This section will outline the flow development simulation, detailing mesh generation, the incompressible PISO solver and modifications made to it, numerical discretization schemes and y parameter computation. Relevant case directory files read by OpenFOAM are discussed in more detail in the appendix including a by a step by step decomposition of the main CFD run shell script.

3.2.1 Mesh

A mesh may be developed in one of two ways namely using the standard OpenFOAM **blockMesh** functionality with a corresponding **blockMeshDict** file, or alternately generating a mesh in an external meshing application and importing it into the CFD environment. The latter was implemented. An external mesher Gmsh was used to produce the mesh files required for import. Different meshes were required under different flow conditions to ensure that the confines of the wall function patching parameters y^+ and y^* were respected. In order to confine the parameters within acceptable bounds a wall graded mesh was used, although this was not necessary in all cases. By varying the size of wall adjacent cells the parameters could be adjusted. The imported mesh boundaries had to be redefined locally using the **createPatch** utility executing commands defined in the **createPatchDict** file of fragment *C.2*. A shell script **mesh.sh** fragment *C.3*, was designed to automate all meshing operations and associated file transfers and file removal. The script procedure comprised sequential meshing, import, patching and cleaning operations.

Once the mesh was imported the **checkMesh** utility was run to produce qualitative mesh statistics including cell skewness and cell numbers. The geometry was cubic and meshed using regular hexa-

hedral elements, hence cell skewness was not expected to be problematic, however it is normally an important indicator of mesh quality. The geometry was built systematically defining points, then lines, surfaces and volumes, refer to fragment *C.4*. Wall grading was adjusted by varying the **bump** factor in line *27*. The **bump** functionality which grades bidirectionally was used on all lines comprising the cross section. This was possible due to the symmetry of the square cross section, otherwise each dimension would have to be graded separately. Surface boundaries were named along with the volume and subsequently redefined in the **createPatchDict**.

The **createPatchDict** file was required to patch the imported mesh correcting undesired internal field assignments and other minor irregularities which were incurred through the import process producing a simulation ready mesh. The entries detailed in fragment *C.2* reveal the nature of the patch with respect to all bounding surfaces including inlet, outlet and walls. OpenFOAM facilitates for a number of boundary types including cyclic, patch, wall, etc, however since a pressure gradient was to be specified, the inlet and outlet patches had to be defined separately in order to set different pressure values at each boundary. Any particular mesh could easily be adjusted by changing a maximum of three numbers in the corresponding geometry file, namely the grading factor for the x and y dimensions and the number of cells in each of the three dimensions, bearing in mind that because of the symmetric cross section two quantities are eliminated, then rerunning the mesh and patch operations to incorporate the mesh into OpenFOAM.

Case study 1 corresponds to an approximate centreline velocity peak of 0.1 ms^{-1}. Mesh projection views are presented in figure *3.4*. The clipped view shows internal mesh regularity. Mesh specifics are also included i.e. total mesh elements with x y z distribution as well as wall grading ratio r. The number of cells in the xy plane and the grading ratio along with flow velocity characteristics determine the values of y^+ and y^*. The need for mesh refinement to retain these dimensionless wall function parameters y^+ and y^* within acceptable ranges i.e. between 30 and 150, under higher flow rates is apparent when considering appendix meshes *A.6* and *A.13* in comparison to mesh figure *3.4*. The grading ratio refers to the ratio between the smallest and largest block size in a given dimension. The analytic relations which describe block size progression may be found in the OpenFOAM user guide [43] pg 39. Mesh density contributes considerably to computation time hence the coarser, grid independent configurations respecting y^+ constraints would produce time optimized simulations. There was no grading in the flow direction, however cell sizing in the flow direction was related to Courant number and hence also to the time step. Finer meshes resolve velocity and other variable gradients better, which propagate throughout the domain resulting in more accurate overall solutions.

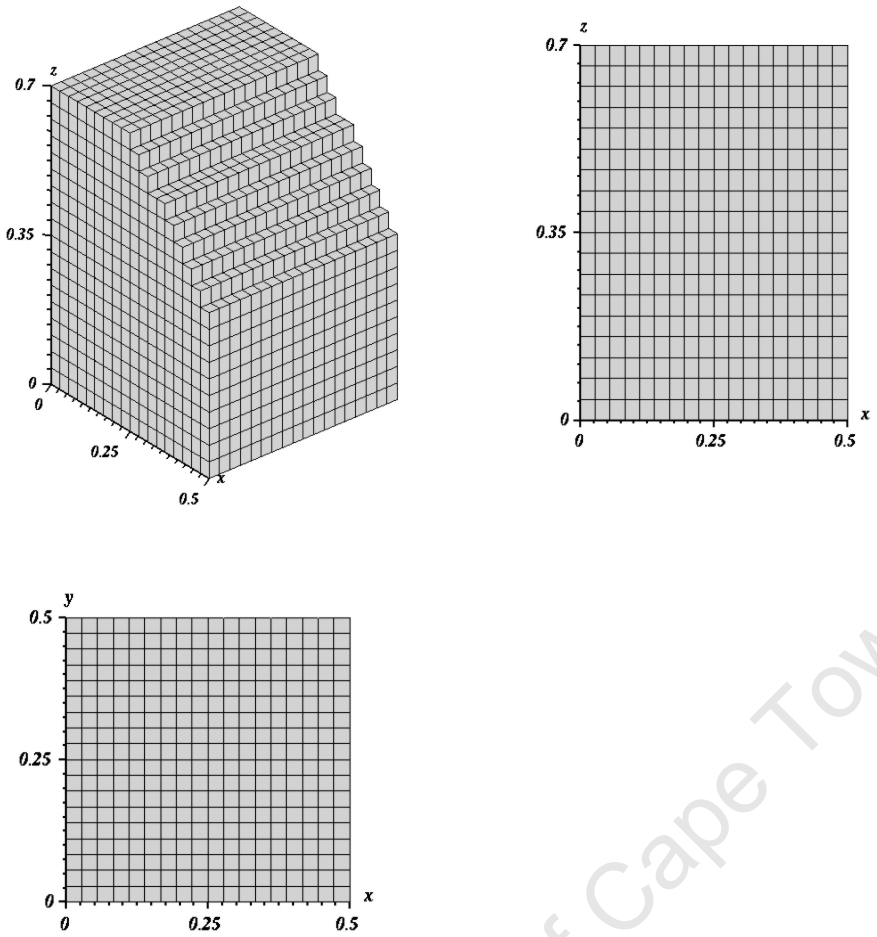


Fig. (3.4) *Mesh elements: 5832, (n_x, n_y, n_z) = (18, 18, 18), no wall grading r=1*

3.2.2 Solvers and Models

The study undertaken implemented a RANS based simulation which essentially sacrifices flow physics for computational speed through the averaging procedures used to derive the governing forms. Compared to LES and DNS models which retain more physics, RANS models do more modelling and less resolving. DNS models are based purely on resolving all scales of flow, and LES models are somewhere in between RANS and DNS. Turbulence is accounted for using a $k - \epsilon$ two equation model based on the Bousinessq hypothesis, the foundation of any eddy viscosity model. As mentioned previously RSM's are better suited for turbulent flow development in ducts but were decided against because of their complexity, potential instability and computational demand, deemed unnecessary for first pass analysis. To the authors knowledge OpenFOAM has two RSM models available, namely the Launder Rodi Reece (LRR) and Launder Gibson RSTM. The RANS equations are solved for average velocity and pressure fields using the PISO solver algorithm as discussed previously. The solver essentially couples pressure and velocity through iterative progression.

Standard PISO Solver Algorithm

The velocity related LHS of the NSE as detailed in the previous chapter is coded in the PISO solver as in fragment *C.5*. The LHS is denoted **UEqn** and is equated to the RHS in the momentum predictor relation which contains the pressure gradient term. The incompressible PISO algorithm does not account for the gravitational contribution on the RHS of the NSE.

The basic steps in the predictor corrector algorithm are detailed in section *2.2.1*. Fragment *C.6* points out the pressure correction equation as well as the flux and momentum corrector equations according to Wierinks description [57]. The nature of the driving components determines the necessary treatment in the PISO algorithm i.e. a spatially variant momentum source contributes to the pressure correction equation whereas a constant momentum source does not as its divergence is zero. A constant pressure gradient in the primary flow direction is therefore equivalent to a constant momentum source or sink in the NSE, hence the inclusion of gravity is not necessary for simulating vertically orientated pressure driven flow as the pressure gradient can be adjusted to offset gravity. It should be noted by inspection of the NSE that driving terms on the RHS become balanced by the opposing viscosity related contribution which is the fluids resistance to motion. The higher the flow velocities the greater the dissipation rate and hence an equilibrium state must be reached where the RHS driving terms become balanced by the LHS velocity function terms.

PISO Adaptive Time Stepping Modification

The PISO solver had been modified to introduce ATS functionality. ATS facilitates optimized computation time and is only relevant in the case of flow development. Integration of this functionality into the coupled framework would require additional coding on both CFD and DEM sides to ensure congruency not only in terms of progressing the simulation but also with respect to write interval as inconsistent output data cannot be viewed simultaneously in ParaView. It should however be noted that in this particular study the converged steady flow solution which is mapped into the coupled framework would be accompanied by a somewhat steady Courant number and hence the time step would not be expected to vary. Under steady flow ATS has no significant advantage, and only utilizes resources to recompute timestep. Furthermore a coupled simulation timestep is in most cases dependent on that required to adequately resolve the DEM solution. Discrete element time steps are considerably smaller than those used in continuum problems. Time increments of the order 10^{-6} s are common [20]. Since the coupled simulation time increment is based on DEM resolution which is typically smaller than that required to resolve the CFD solution, the Courant number will be maintained low and cannot be used as a measure upon which to adjust time increment.

The standard **setDeltaT.H** header file provides the desired ATS functionality needed to develop the steady flow profile. In its standard form the header accounts for two phases as it was coded for implementation in the multiphase solver **interFoam**. Inclusion of **.H** files in the body of a **.C** file in OpenFOAM is equivalent to pasting the corresponding section of code in its place. The **interFoam** header file **setDeltaT.H** was modified to exclude all multiphase terms, the corresponding code fragment *C.7* is appended. The code fragment *C.7* factors the current time step based on the user

specified Courant number limit and the current computed Courant number field. An initial timestep had to be specified which was then adapted as the solution progressed. The modified PISO solver was named **pisoFoamATS**, with ATS appended to denote the functionality.

3.2.3 Discretization Schemes

Two major components of any CFD code include numerics and physical modelling. Equations which describe physical behaviour are related to physical modelling whilst numerical schemes used to solve the equations fall under the numerics constituent. Governing equations are rarely solvable through analytic means and hence have to be approximated through implementation of numerical methods. Governing equations may contain full or partial derivatives, time derivatives, spatial derivatives or gradients, divergence terms, double derivatives of time and space and others. Numerous numerical schemes have been devised over the years to approximate each type of variant physical behaviour denoted mathematically by these derivatives, gradients, etc. OpenFOAM contains a number of these schemes which gives more freedom to the user in setting up a numerical system as no single method has proven universally superior to all others. The choice of method depends on the desired result with respect to stability, accuracy and computation time. Often methods work in conjunction and have to be implemented in unison. Numerical schemes are specified in the **fvSchemes** file fragment *C.9*. Specifics and relevant equation forms pertaining to numerical schemes can be found in the OpenFOAM user guide pg 113-125 [43] and in the OpenFOAM programmer guide pg 37-44 [42]. A detailed treatment of simulation numerics is beyond the scope of this study and will not be commented on in depth. Scheme adjustment is more suited to complex models comprising irregular geometry, meshing and elaborate boundary conditions.

The schemes implemented in this study are those in fragment *C.9* also tabulated in *3.4*. A first order, bounded, implicit Euler time scheme was used. Other schemes include **Crank Nicholson** and **backward differencing**. **Crank Nicholson** is a second order, bounded, implicit scheme which is more prone to instability and can be adjusted by an associated coefficient which blends it with the more stable Euler scheme. The time derivative typically applies to that of velocity in the NSE. Gradient schemes are all Gaussian types with linear interpolation. Gaussian integration is a second order method and implements its own interpolation scheme to compute face centred values from cell centred ones. The need for interpolation arises from the fact that Gaussian integration is based on summation of face values. Linear interpolation is effective in most cases. One such example of gradient operation pertains to the pressure field in the NSE. Other gradient schemes supported in OpenFOAM include **leastSquares** and **fourth**. Defaults which are set apply to all such operations executed during the solution process which have not been explicitly assigned.

A variety of divergence operations are carried out in solving turbulent averaged NSE, introduced in the additional $k - \epsilon$ equations of the RANS **kEpsilon** model. The definitions are provided under the keyword **divSchemes**. The Gaussian scheme is the only option in this case for which a variety of possible interpolation schemes can be selected, most of which have been chosen as linear or linear limited 1. Convection specific schemes such as **limitedLinear** are known as Total Variation Diminishing (TVD) schemes and compute interpolation based on flow velocity flux. The corresponding coefficient

of unity in all divergence operations which is generally recommended, is specified for best convergence. Divergence scheme selection has considerable effect on numerical behaviour.

Laplacian schemes require both interpolation and surface normal gradient schemes to compute, the template form is provided in line *22* fragment *C.9*. The only option is Gaussian and the scheme is used to compute, amongst other terms the rate of deformation tensor in the NSE. Surface normal gradients, at cell faces, are necessary to compute Laplacian terms. A surface normal gradient is that gradient of some variable, in the direction of the surface normal at a face connecting two adjacent cells. The default scheme **corrected** refers to an explicit non-orthogonal correction method.

The **fvSolution** file was used to specify solver behaviour. The solvers defined here are those which perform the actual matrix computations, whereas previously defined solution algorithms are used to populate matrices and vectors. NSE are typically solved for pressure and velocity, whilst turbulence related fields are solved for through an additional set of equations. There are a number of matrix equation solvers which may be selected from in OpenFOAM, the most common being the Preconditioned Conjugate Gradient (PCG) Solver for symmetric matrices and the Preconditioned Bi-Conjugate Gradient (PBiCG) Solver for asymmetric matrices. Matrix equations are often preconditioned through decomposition or factorization methods including Diagonal Incomplete Cholesky (DIC) for symmetric matrices and Diagonal Incomplete Lower Upper (DILU) for asymmetric matrices. Preconditioning produces numerically workable matrices which can be solved more efficiently. Preconditioning operations and solvers are complimentary hence symmetric solvers must be accompanied by symmetric preconditioning evident in fragment *C.10* lines *5* and *6* for a symmetric pressure solver and lines *14* and *15* for asymmetric velocity. The asymmetric form is adopted by all other relevant field variable solvers including **k**, **epsilon**, **R** and **nuTilda**. Additional solvers including smooth and Generalized Geometric-Algebraic Multi-Grid (GAMG) types are supported, however PCG type solvers are common and reliable and thus implemented in the study. All solvers are based on reducing residual between successive solution iterations. Smaller residuals are indicative of higher degrees of solution accuracy. The iteration procedure completes at a specified residual tolerance typically of the order of 10^{-5} or 10^{-6}. Additional solution limiters exist, either based on relative tolerance i.e. the ratio between current and initial residuals or on the maximum number of iterations or both, whichever comes first. Zero relative tolerance indirectly forces the solver to converge the residual at the specified tolerance level.

PISO algorithm parameters are set as in lines *24* to *27* fragment *C.10*. The PISO algorithm evaluates an initial solution and then corrects it based on the number of orthogonal corrections specified in line *24*, typically not more than four in transient cases. The number of orthogonal (velocity) and non-orthogonal (pressure) corrections to be applied are dependent on the mesh. The current study deals with a fully orthogonal mesh i.e. a mesh in which connecting face normals are parallel to vectors connecting cell centres, hence non-orthogonal corrections are set to zero. Otherwise the number of non-orthogonal corrections is set in relation to the degree of non-orthogonality of the mesh. The reference values for pressure in lines *26* and *27* are generally related and implemented as needed in closed systems where relative pressure is important and not the total value of pressure. Since the pressure gradient is explicitly defined in the study to produce various flow velocities, the importance

of the pressure solution and its relativity is auxiliary. Relaxation factors may be applied to improve computational stability by limiting the amount by which a variable may change between successive iterations.

The discretization schemes and solvers implemented in the model are tabulated in *3.4*. All discretization schemes used were standard, based on OpenFOAM tutorials. These schemes are not defaulted ad hoc but have been found to produce best results on average for the widest range of simulation problems. Specialized treatment would be warranted if simulation stability becomes questionable, which was not the case in this study. The $k-\epsilon$ two equation model is know to be stable. In addition the mesh was regular and errors could not be incurred through poor element aspect ratios. In the authors opinion there was no need to investigate this area further considering the main aim of the study was not solely the computational fluid model. The grayed rows are schemes which have to be included under coupled simulation to account for momentum transfer between the particles and fluid.

Tbl. (3.4) *Discretization schemes and solvers*

quantity	solver	tolerance	$\boldsymbol{\nabla}\cdot$ (*Gauss* -)	$\boldsymbol{\nabla}^2$ (*Gauss* -)
p	PCG	$1\,e^{-6}$	–	–
$\mathbf{U}$	PBiCG	$1\,e^{-5}$	–	–
k	PBiCG	$1\,e^{-5}$	–	–
ϵ	PBiCG	$1\,e^{-5}$	–	–
$\mathbf{R}$	PBiCG	$1\,e^{-5}$	*linear*	–
$\tilde{\nu}$	PBiCG	$1\,e^{-5}$	–	–
$(\phi\mathbf{U})$	–	–	*limitedLinearV 1*	–
(ϕk)	–	–	*limitedLinear 1*	–
$(\phi\epsilon)$	–	–	*limitedLinear 1*	–
$(\phi\mathbf{R})$	–	–	*limitedLinear 1*	–
$(\phi\tilde{\nu})$	–	–	*limitedLinear 1*	–
$\left(\nu_e\,\mathrm{dev}\boldsymbol{\nabla}U^T\right)$	–	–	*linear*	
$(\nu_e\,\mathbf{U})$	–	–	–	*linear corrected*
$(k_e\,k)$	–	–	–	*linear corrected*
$(\epsilon_e\,\epsilon)$	–	–	–	*linear corrected*
$(R_e\,\mathbf{R})$	–	–	–	*linear corrected*
$(\tilde{\nu}_e\,\tilde{\nu})$	–	–	–	*linear corrected*
$(A(\mathbf{U})\,p)$	–	–	–	*linear corrected*
$\left(visc\,\mathrm{dev}\boldsymbol{\nabla}U^T\right)$	–	–	*linear*	–
$(visc\mathbf{U})$	–	–	–	*linear corrected*
$(\alpha_f\,A(\mathbf{U})\,p)$	–	–	–	*linear corrected*

$\frac{\partial}{\partial t}$ default *Euler*
$\boldsymbol{\nabla}$ default *Gauss linear*
interpolation default *linear*
surface normal gradient default *corrected*
visc viscous term

PISO orthogonal correctors 2
PISO non orthogonal correctors 1
$\phi = \rho\mathbf{U}$
$k_e = \frac{\nu_t}{\sigma_k}$
$\epsilon_e = \frac{\nu_t}{\sigma_\epsilon}$

3.2.4 Boundary Conditions

Near Wall y^+ and y^* Parameters

The primary value of implementing wall functions in turbulent simulations is to achieve a marked reduction in computation time through facilitating the use of considerably coarser near wall meshing. Details relating to wall functions have been outlined in section *2.2.1* and will not be repeated here. Suffice to say that significant mesh refinement would otherwise be required to resolve the steep velocity gradients at the walls. Wall functions thus model wall gradients and the bulk mesh resolves gradients. Zero slip walls are assumed in this study. OpenFOAM contains post processing utilities which compute y^* for RANS type simulations and y^+ for LES type models. The **yPlusRAS** utility produces a y^* field based on equation *2.2.7* and extracts minima, maxima and field averages. The corresponding code fragment *C.11* is extracted from the OpenFOAM **RASModel.C** file.

A user defined function which computes both y^+ and y^* parameters was implemented to provide validation results against **yPlusRAS** output. The user defined function fragment *C.12* was based on equations *2.2.6* and *2.2.7*, section *2.2.1*. Quantities required to compute the equations, including wall velocity gradient line *5*, wall turbulent kinetic energy line *10* and wall turbulent viscosity line *15* are extracted from output files produced through sampling operation.

The user defined code is called at interval to produce y^+ and y^* data as the simulation progresses. **yPlusRAS** is called at the end of the simulation to produces y^* history data at the same time intervals as the user defined function, facilitating comparative measure. In addition y^+ and y^* values should be comparable when the wall adjacent cell centre is located within the logarithmic layer of the physical flow. This serves as an additional system check. It should be noted that this type of user coded analysis is greatly simplified by the regular mesh geometry used in this study.

3.3 DEM/CFD-DEM Model

The following sub-sections are dedicated to the LIGGGHTS DEM configuration. The coupled modified PISO algorithm is detailed, followed by an overview of the modifications made in LIGGGHTS to introduce the acoustic force, as well as secondary modifications relating to unit conversion and data output. Stepwise dissections of the LIGGGHTS input and coupled calling scripts are appended along with a discussion on the additional files required for the coupled simulation.

3.3.1 Coupled PISO Solver Algorithm

The coupled solver **cfdemSolverPiso** inherits the same basic structure from the parent PISO algorithm with a few equation modifications to account for the momentum exchange terms which manifest through fluid interaction with the discrete particulate phase. An **evolve** step in included at the beginning of the PISO algorithm as in fragment *C.23* which calls the CFDEM solver to compute all fluid particle exchanges and produce the relevant fields which are required in subsequent computations.

The scalar $\mathbf{K}_{sl}$ field along with the particulate velocity field $\mathbf{U}_s$ are introduced into the NSE as in fragment *C.23*. The fluid particle momentum exchange field $\mathbf{R}_{sl}$ defined in section *2.2.3* is separated into two terms for numerical convenience. The fluid velocity related component being fed to the LHS of the NSE and the particulate velocity related component being fed to the RHS. The turbulent term accounting for Reynolds and deformation stress tensors is replaced by **divVoidfractionTau** which is equivalent in form to that of the standard formulation but with the inclusion of a void fraction multiplier α_f. Void fraction premultiplies all terms in the NSE either directly or indirectly as in the case of the formulation of $\mathbf{K}_{sl}$ where void fraction is accounted for intrinsically. In dilute systems or systems containing very small particles, the void fraction would be near unity i.e. the fluid fraction would be dominant, which would result in partially recovered NSE.

The $\mathbf{K}_{sl}$ term is divided by density and the resulting term is denoted by $\mathbf{K}_{sl,\rho}$ in fragments *C.23* and *C.24*. It should also be noted that the pressure term in the incompressible NSE is kinematic. All particulate phase fields including $\mathbf{K}_{sl}$, $\mathbf{U}_s$ and α_f are discrete. The particulate velocity subscript **p** used in the previous chapter has been replaced by **s** to resemble more closely the naming convention implemented in the code. There are two forms of the coupled NSE, differing with respect to the treatment of void fraction. The two models are denoted by **A** and **B** respectively. Model **B** was selected for the study as the effect of dilute system void fraction would be negligible. In addition it is the favoured model used by the developers in all package tutorials, hence all corresponding equations in the documented code fragments relate to model type **B**.

The RHS discrete source component of momentum exchange has to be dealt with separately to the fluid velocity related component. The additional source term brought about by the particulate phase is incorporated into the Pressure Correction Equation through interpolated surface scalar fields. Again it should be noted that constant momentum terms do not form part of the Pressure Correction Equation as their divergence is zero. Details of the interpolation and subsequent derivation of the pressure equation will not be detailed. It is suffice to say that vector and scalar fields have to be interpolated to faces and vector fields require subsequent contraction with surface vectors to produce the flux fields of the Pressure Correction Equation.

The Momentum and Flux Correction formulations implemented in the modified PISO algorithm and documented in fragment *C.24* are of similar form to those implemented in the standard PISO algorithm but modified slightly to account for the added momentum produced through fluid particle interaction. The momentum term which had been designated to the RHS of the NSE as a function of particulate phase velocity is treated in the velocity and pressure correction loops. It should be noted that the specified number of orthogonal corrections is increased to achieve convergence. This may be attributed to the discrete nature of the velocity field post interaction. In dilute systems and in systems with very small particles the discrete phase contribution to the momentum equations is negligible and effectively one way coupling is realized. The CFDEM code facilitates one way coupling however the two way MPI coupling method was implemented in this study. This method is commonly used and is essentially equivalent to one way coupling when dealing with a particulate phase of low momentum contribution, however at additional computational cost.

In the current study a constant vertical pressure gradient was maintained across the control volume.

If gravity were to be included in the standard incompressible PISO algorithm the pressure gradient could be adjusted accordingly to account for it, as no specific pressure gradients were prescribed for this study. The inclusion or exclusion of gravity was not expected to change the flow profile because of its vertical orientation. The gravitational contribution may become relevant if a pump were to be specified to produce a certain flow velocity.

3.3.2 LIGGGHTS Modifications

Acoustic Force

A separate LIGGGHTS function known as a fix was coded to account for the acoustic force. The code was derived from previously existing fixes standard with the LIGGGHTS release used in the study. The code implemented in LIGGGHTS to account for the acoustic force on suspended particles is detailed in fragment *C.25*. The acoustic function fix computes unique forces on each particle according to equations detailed in section *2.1.3*, presented alongside relevant lines of code. The user may specify the direction in which the acoustic force is to act. A number of these fixes can be applied simultaneously in the case where transducers act perpendicular producing interfering resonant modes in a regular shaped cavity. The formulation of the force is limited to rectangular geometry. Most parameters are computed inside the code itself however certain parameters including driving amplitude, frequency, particle modulus, fluid density and fluid compressibility have to be passed as arguments into the function. The reason for argumenting fluid parameters is to allow the user to specify the carrier fluid. In addition the user may wish to vary acoustic parameters and the direction in which the force acts. The function also allows the user to select a specific group of particles or geometric region to which or in which the acoustic force is applied.

Dump Output

In order to produce matrix type output which could be easily imported into a post-processing environment, the standard LIGGGHTS dump code **xyz** was modified to output additional information apart from only the particle coordinates in raw type format. Any file writing output is referred to as dump operation in LIGGGHTS. A number of dump formats are standard with the LIGGGHTS release however, none were found to produce the desired output in raw format. Typical output formatting included text and irregular data structures. Furthermore particle data is not output systematically under parallel computation i.e. particle data is not written in order from first to last particle at every write interval but rather in an erratic fashion. Unpredictable output is not easily extractable for plotting and data analysis purposes. The **xyz** function on the other hand does maintain output structure by buffering and allocating space for specific particle entries. The code was extended to include additional output data which could be useful not only in this study but in future studies as well. The output format is detailed in fragment *C.26*.

CGS to SI Conversion

The need to simulate small particles of the order 10^{-5} m, was facilitated by using a Centimetre Gram Second (CGS) unit system. Implementing the International Metric (SI) unit system in the DEM environment would produce machine precision errors when dealing with volume quantities, of cubic proportionality to particle size. LAMMPS had been designed to support a variety of unit systems whilst OpenFOAM had been designed to support only SI and Imperial unit systems. Coupling of the two codes would produce dimensional inconsistency since the MPI only sends values across. A unit conversion scheme such as that tabulated in *B.4* was required to maintain dimensional consistency. The necessary unit conversion could be applied at various stages of the coupling process. The standard coupling code passes values of particle velocity, radius and position from LIGGGHTS to OpenFOAM. The CFDEM OpenFOAM extension code then computes fluid related particle forces which are fed back to LIGGGHTS, and also momentum exchange terms to be used in the solution of the modified NSE, refer to figure *3.3*. A fellow researcher Evan Smuts Centre for Research in Computational and Applied Mechanics (CERECAM) University of Cape Town (UCT) provided modified code to produce the necessary conversions. The proposed solution inserts conversion factors at root level functions which are subsequently called to exchange vector and scalar values between LIGGGHTS and OpenFOAM. The low level conversion implemented in fragment *C.27* may become undesirable if additional functionality is introduced into the coupled solver requiring additional quantities to be passed which do not have the same conversion factors, however the solution provided by Evan proved adequate for this study. To circumvent this possible shortcoming however, conversion factors could be implemented at higher levels such as in force and momentum transfer model codes themselves thereby localizing their influence.

In order to produce dump output in a metric system, consistent with that of OpenFOAM, for simultaneous CFD DEM viewing in ParaView, an additional conversion code had to be introduced based on the standard dump code. Fragment *C.28* was extracted from code supplied by Evan Smuts CERECAM UCT to apply appropriate conversion factors as in lines *6*, *14*, *24*, *34*, etc for various quantities. The fragment does not contain all entries to which conversion factors had been applied in the interest of brevity. Conversion with respect to particle position and size was necessary for visualization purposes however additional conversion may be required if post processing analysis is to be conducted inside ParaView. For the purposes of the current study position and size conversion was sufficient as no further analysis was done in ParaView.

3.4 Model Validation

The CFD model was more conducive to validation because of its regular geometry and an abundance of experimental data. There were four methodologies used to validate the CFD model, namely the y parameter check, grid independence resolution, the pressure drop to mean velocity relation in the form of the Darcy-Weisbach equation *2.2.9*, and a qualitative evaluation of the resulting profiles.

The coupled CFD-DEM model was not conducive to validation because all large scale chamber studies

found in literature were primarily concerned with agglomeration. The current model was not set up to model inter-particle forces responsible for cohesion between particles. In addition any filtration system is evaluated on filtration efficiency, which would require the inclusion of particle collisions because these essentially deter optimal performance i.e. a dense suspension may respond poorly to acoustic force filtration due to a number of factors, including acoustic scattering and inter-particle collisions. Filtration systems are typically evaluated by model and experiment. Small scale chambers can rely primarily on the acoustic force and a laminar carrier flow to achieve some degree of fluid clarification, however larger scale systems, implementing the acoustic force to produce agglomerations depend on additional processes downstream to perform the final filtration, the outputs of which can then be quantified through efficiency percentages. The current model does not include any staged filtration mechanisms as these would require additional physics modelling, which was beyond the scope of this study. The only plausible method of evaluating the acoustic force and particle model would be qualitative, by investigating general trends and comparing them with relevant governing equations, observations made by published researchers and intuitive expectations. It should be noted that system validation is different to isolated force model validation. Systems are coupled hence unexpected results can emerge from physical or numerical phenomena, whereas simple force models, which are less prone to numerical error have analytic forms which should be recovered if simulated in the absence of disruptive forces.

Table *B.5* lists the expected analytically derived periodic nodal positions for the ideal standing wave pressure field under various driving frequencies. Nodal x coordinates were computed through equation *2.1.8*. The table may be used as a comparative instrument to test simulation adherence to the analytic solution model i.e. particles are expected to be forced to these nodal coordinates. The table is supplemented by graphical representations figures *A.5 a* to *f* of pressure and acoustic force fields, similar to figure *2.1* of section *2.1.3*, for a 35 micron diameter particle, under various driving frequencies, keeping outstanding parameters constant. Force is shown on the left axis and pressure on the right. The acoustic force was computed through equation *2.1.6*. It may be noted from graphical and tabulated data that both the number of pressure nodes and acoustic force peak amplitudes increase with driving frequency. Pressure amplitudes registered at anti-nodal planes remain unaffected by driving frequency.

CHAPTER 4

RESULTS & DISCUSSION

This chapter is split into two sections, dealing with flow development and particulate response behaviour respectively. Under flow development, validation will be presented followed by results pertaining to flow and turbulence profiles. We introduce the coupled CFD-DEM simulation by discussing numerical and physical validation. Trajectories are analysed for particles of various sizes and densities transported at three different flow rates passing through fields of two acoustic driving frequencies. A further analysis of the driving acoustic and drag forces follows. Some numerical technicalities are also discussed.

The results presented in the flow development section are those pertaining to case study 1, centreline velocity 0.1 ms^{-1}. Similar procedures were implemented for all other case studies which produced similar trends, the results of which are appended in A and B and referred to as necessary when comparisons are made. A summary list of pertinent findings concludes the section, along with tables consolidating simulation time and result data for all flow development runs. Subsidiary plots, tables and figures which are not referred to in the main text, but considered useful are appended for reference purposes and for the sake of completeness. Certain tables and plots are not appended as they were perceived to contribute little toward the primary objectives of the study. Section *4.2* dealing with the particulate system will focus on case studies 1 and 2, centreline flow velocities 0.1 and 0.5 ms^{-1} with appended data for case study 3. The section will conclude with a summary list of pertinent findings and tabulated simulation time statistics.

4.1 CFD

4.1.1 Validation

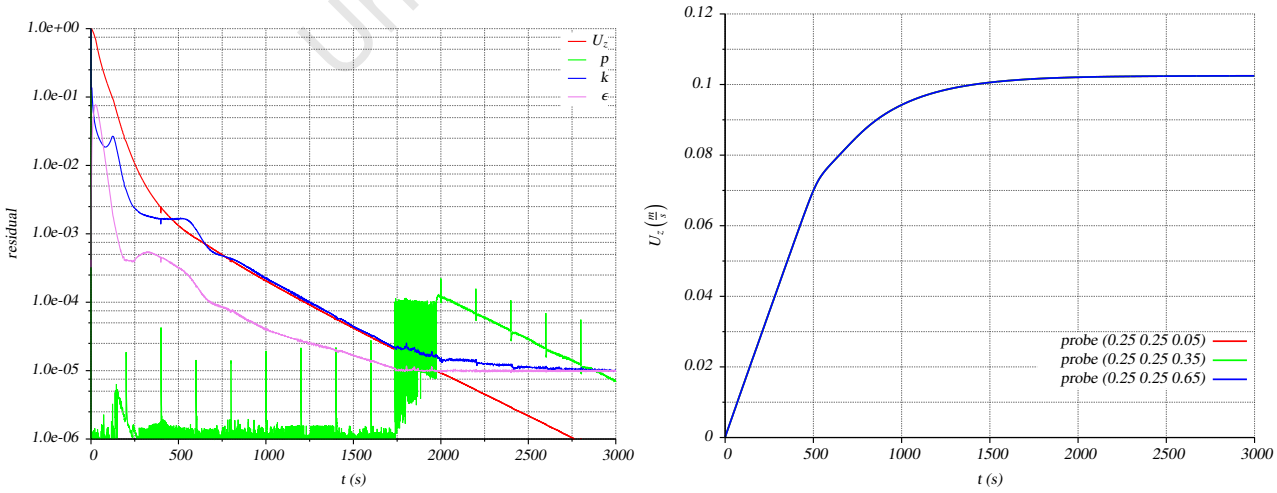
This section will deal with flow validation. Four methods were employed, including residual analysis, grid independence, y parameter checks, log of the wall profiles and validation against analytic prediction, as outlined in section *3.4*.

Convergence

Figure *4.1 a* is a plot showing residual decline against simulation time for a number of pertinent variables. Residuals level off at around 10^{-5} or 10^{-6}, some with negative gradients at termination indicating a trend toward even lower residuals and further convergence. The residual values correspond somewhat to those specified in the **fvSolution** file. This residual range was deemed acceptable for the purposes of this study. A pressure residual jump is evident around $t = 1800\ s$, which was attributed to algorithmic predictor corrector response at solution convergence. This characteristic pressure residual jump at near convergence was noted in all cases, refer to appendix figures *A.7* and *A.14*. The pressure residual subsequently falls, also noted in all case studies.

Figure *4.1 b* shows the z component of velocity, sampled (probed) at various heights along the duct centreline, against simulation time. Sampled velocities are seen to overlap from the onset of simulation indicating that there were no velocity gradients in the z direction at any time during the simulation run. The overlap and general convergence form was apparent in all simulated cases. Case studies were distinguished based on these probed velocity maxima and not on average flow velocities. These probe based plots, updating during run time were used to iteratively adjust pressure gradients across the control volume to obtain the required flow velocity.

A clear correlation can be seen between falling residuals and a decrease in velocity gradient, bearing in mind the exponential decline of residuals. There was clear correlation between the time at which the pressure jump was noted and that at which probed velocities attain a near zero gradient. Convergence time was seen to be inversely proportional to flow rate as noted by comparing convergence data of all three cases. This was attributed to entrance length phenomena i.e. that duct length required to develop a steady turbulent flow profile from initial conditions. The distance travelled by the fluid over a given period of time was dependent directly on flow velocity which was in turn driven by the applied pressure gradient. Higher pressure gradients produce faster flows which cover the entrance length in shorter periods of time.



(a) Residuals *(b) Probed velocity*

Fig. (4.1) *Solution convergence*

Grid Independence and Near Wall Profiles

Figure *4.2* shows z velocity component profiles in the xz plane obtained for different mesh densities. Averaged velocities denoted by corresponding dashed lines were obtained by integrating surface profiles. The grid was subsequently refined from 512 to 17576 cells corresponding to 8 and 26 cells in each Cartesian direction respectively. It is evident from the figure that the velocity profile in the bulk flow region i.e. between 0.1 and 0.4 m is resolved at a fairly low level of refinement around 14 cells, whereas the peripheries require additional elements to resolve the flow near the walls. No wall grading was implemented and perhaps the near wall profiles would have been better resolved by coarser meshes had they been graded toward the wall boundaries. Since the particles are inserted in the bulk flow between 0.15 and 0.35 m the near wall profiles are of secondary importance when considering the influence of the flow field on the particles. Thus we selected the $18 \times 18 \times 18$ element configuration for case study 1.

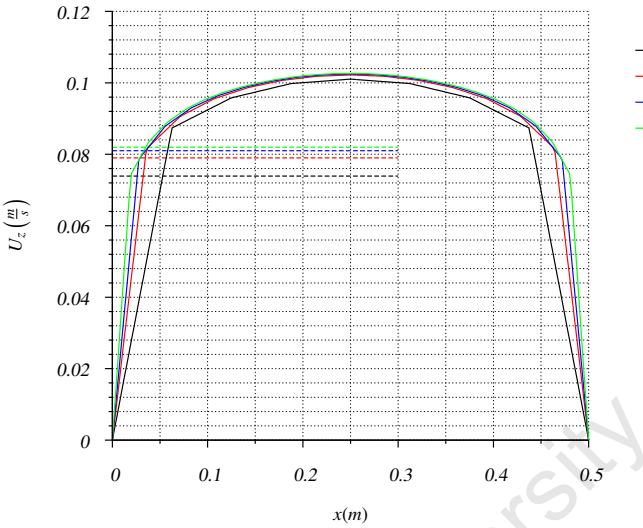


Fig. (4.2) *Grid independence*

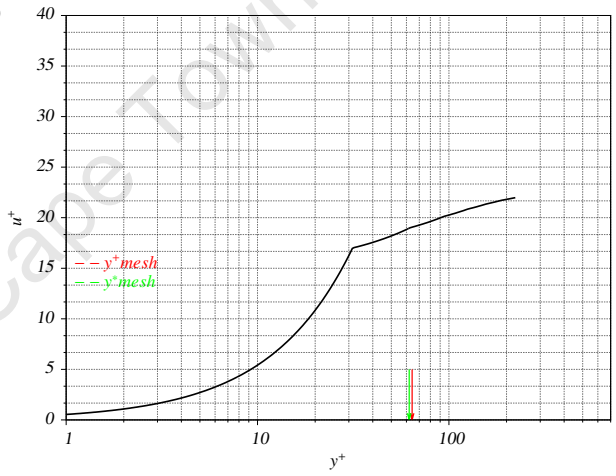


Fig. (4.3) *Log of the wall profile*

Figure *4.3* shows the near wall dimensionless z velocity component profile as well as mesh related y^+ and y^* values denoted by colour coded arrowheads. The plot corresponds to the standard empirically derived log of the wall relation, characterized by a tangential discontinuity at $y^+ = 30$, which describes general turbulent near wall boundary profiles through dimensionless relations. The tangentially discontinuous profile comprising two separate analytic forms was generated by the wall functions implemented. Similar plots are produced in case studies 2 and 3 figures *A.11* and *A.18*. The corresponding velocity data was obtained along a line segment normal to the wall at sampling points which were more frequent near wall becoming more dispersed moving away from the wall. The sample line segment ($x \in [0 : 0.2], y = 0.25, z = 0.35$) comprised 384 points. Code implementation was discussed in section *3.2.4*. The values computed through user introduced functions were compared with those produced by the in-house **yPlusRAS** utility and were found consistent in terms of y^*. It was also found that y^+ and y^* values were comparable in the logarithmic region, implied by closely spaced arrowheads, as was the case in all subsequent simulations, refer to summary data table *4.4*, section *4.1.4*. The y parameters were found to be between 62 and 112 considering all cases indicative

of valid wall function patching.

Table extract *4.1* serves to show the reader the process used to compute boundary layer thickness. The surface ($x \in [0:0.5], y \in [0:0.5], z = 0.35$) z velocity component profile is splined and numerically integrated to obtain its average value, known as the free stream velocity, refer to column 3 of the table extract and also figure *4.2*. The average velocity is then factored to obtain its 99% value. A line profile such as the one depicted in figure *4.2* is then interpolated to find the boundary layer thickness i.e. that distance normal to the wall corresponding to the 99% value of free stream velocity. Boundary layer thickness was not only computed based on integrating the surface profile, different methods integrating line and average profiles are presented comparatively in appendix table *B.6*, however the method outlined proved to be most accurate.

Boundary layer thickness was found to be around 3 *cm* in all cases, refer to summary table *4.4*, computed using method d, table note of *4.1* i.e. numerically integrating splined surface profile data to obtain the free stream velocity. Boundary layer thickness results were mixed and inconclusive, when used to compare cases, revealing no clear trends, possibly attributed to either grid and sample point spacing or to the interpolation methods used both internally within OpenFOAM and externally in user coded post processing functions. Profile integration proved valuable in obtaining mean flow velocities which were subsequently used in validating simulated flow against the Darcy-Weisbach equation.

Tbl. (4.1) *Boundary layer thickness, table extract*

[d] (101×101)	U^{xy}_{max}	0.10246	U^{xy}_{ave}	0.080656	U^{xy99}_{ave}	0.07985	$\delta_{99}(U^{xy99}_{ave})$	0.031424

[d] Splined Surface Node data (corresponding product = number of spline points in each dimension used to integrate surface profile)

Comparison with Physical data

Figure *4.4* shows the Darcy-Weisbach relation based on equation *2.2.9*, between pressure drop and mean flow velocity in turbulent duct flow, for a number of friction factors, over a selected range of mean velocity. Smooth wall friction factors f_D were selected based on average flow Reynolds numbers using the Moody chart. Simulated values of pressure drop and mean flow velocity from table *4.2* are marked on the plot. It is evident from the figure that the pressure drops and corresponding mean velocities obtained through simulation are also obtained analytically. The analytic form was derived on empirical grounds hence is representative of reality. In addition the capped profile obtained through simulation is noted as characteristic of turbulent duct or pipe flow in all textbooks.

A discrepancy was noted between mean and maximum flow velocity ratios predicted by equation *2.2.10* and those obtained using simulated values. These differences were attributed to the values of friction factors used, which were in this case questionable when contrasted with Frank White's textbook, [55] tabulating ratios of 0.811 and 0.849 for Reynolds numbers 10^4 and 10^5 respectively. By adopting these ratios we would see more agreement with those obtained from simulated values.

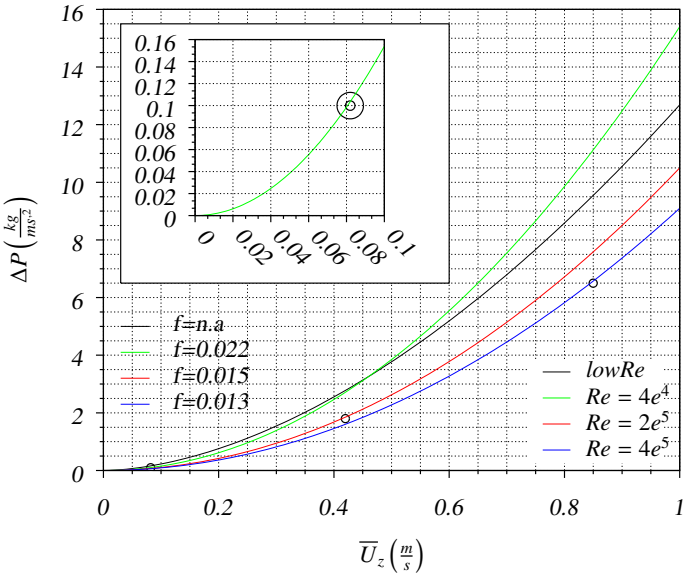


Fig. (4.4) *Pressure drop vs mean velocity*

Tbl. (4.2) *Flow development data*

$\Delta P\left(\frac{kg}{ms^2}\right)$	$V_{max}{}^{a,b}\left(\frac{m}{s}\right)$	$V_m{}^{a}\left(\frac{m}{s}\right)$	$Re_{(V_m)}$	f_D
0.1	0.102	0.081	$4.0\,e^{4}$	0.022
1.8	0.497	0.416	$2.07\,e^{5}$	0.015
6.5	0.997	0.847	$4.23\,e^{5}$	0.013

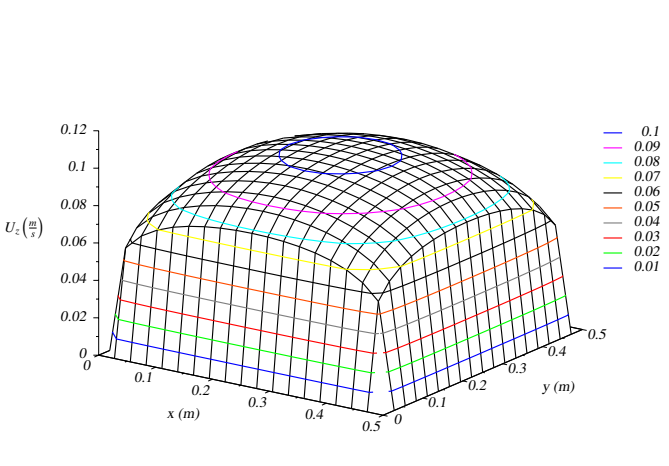
[a] maximum (V_{max}) and mean (V_m) velocities obtained from simulation

[b] exact velocity at the duct centreline obtained from simulation

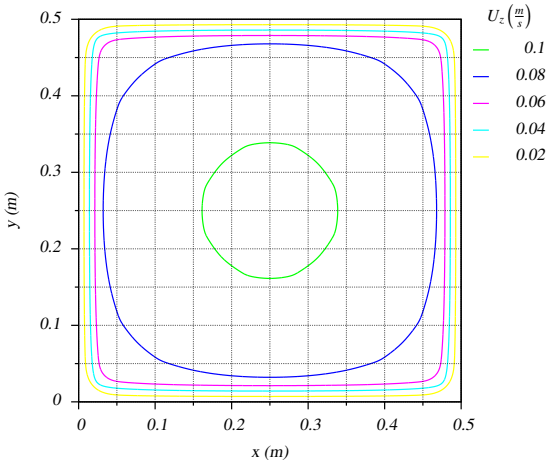
4.1.2 Velocity and Pressure Profiles

Figure *4.5 a* shows the z velocity component cross sectional surface profile constructed from mesh node values sampled at $z = 0.35\ m$. Contour lines have been superimposed to provide further resolution. The capped flow profile characteristic of wall bounded turbulent flows is clearly illustrated. Mesh refinement shows improved velocity profile resolution, refer to appendix figures *A.8 a* and *A.15 a*. Visual representations, also indicative of solution accuracy are enhanced using finer mesh grids.

Figure *4.5 b* is a cross sectional contour plot allowing wall boundary and other gradients to be better visualized. The plot can give some indication of boundary layer thickness provided free stream velocity is known. The cross sectional profile depicted in figures *4.5 a* and *b* remains constant along the height of the simulation domain, except at the inlet and outlet boundaries where wall gradients are not accurate due to the imposed pressure conditions.



(a) *U_z vs xy, mesh grid*



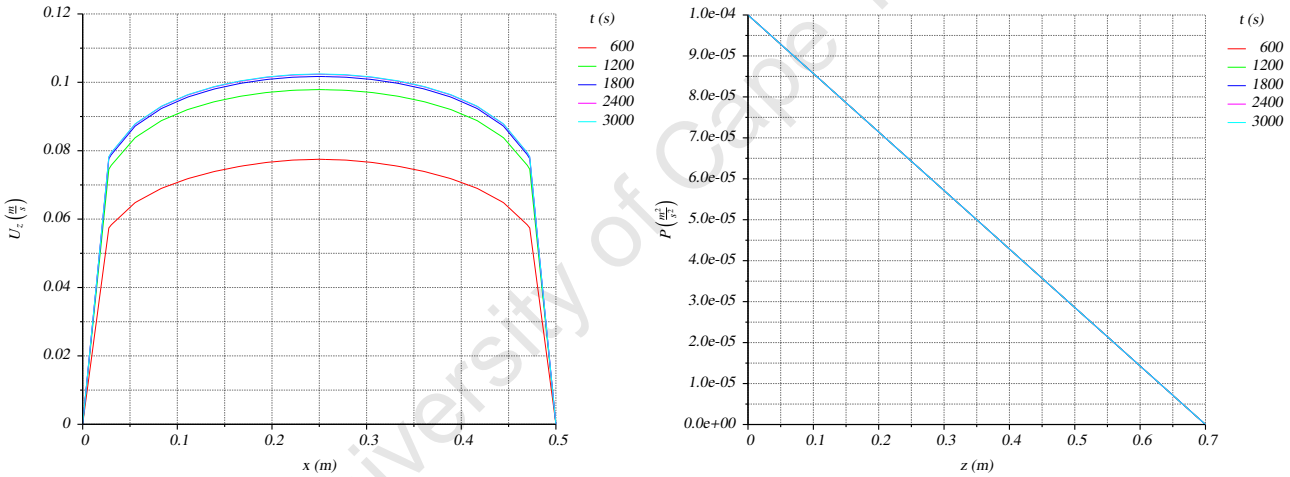
(b) *U_z vs xy, contours*

Fig. (4.5) *Cross sectional flow profiles*

Figures *4.6 a* and *b* show velocity and pressure profile evolution with simulation time. Velocity profiles are sampled along line segments ($x \in [0:0.5], y = 0.25, z = 0.35$) and ($x = 0.25, y \in [0:0.5], z = 0.35$), pressure profiles along line segment ($x = 0.25, y = 0.25, z \in [0:0.7]$), at selected times during the simulation run.

Velocity profiles on xz and yz planes are equivalent because of domain symmetry. Velocity profiles in the yz plane are not documented but had been tested for completeness. It is evident from figure *4.6 a* that the flow profile converges as successive line plots are sampled at equal interval. This convergent behaviour is also noted in case studies 2 and 3, figures *A.9 a* and *A.16 a*, all corresponding qualitatively with respective probe velocity data trends. Near wall velocity gradients, generated by wall functions are steep as suggested in literature.

A constant pressure gradient, as seen in figure *4.6 b* is imposed along the height of the domain for the duration of simulation. Larger pressure gradients are required to produce higher flow rates as may be noted by referring to table *4.4* or to comparative figures in the appendix. Steeper pressure gradients produce faster flows accompanied by steeper near wall velocity gradients.



(a) *U_z vs x* ***(b)*** *P vs z*

Fig. (4.6) *Velocity and pressure profile evolution*

4.1.3 Turbulent profiles

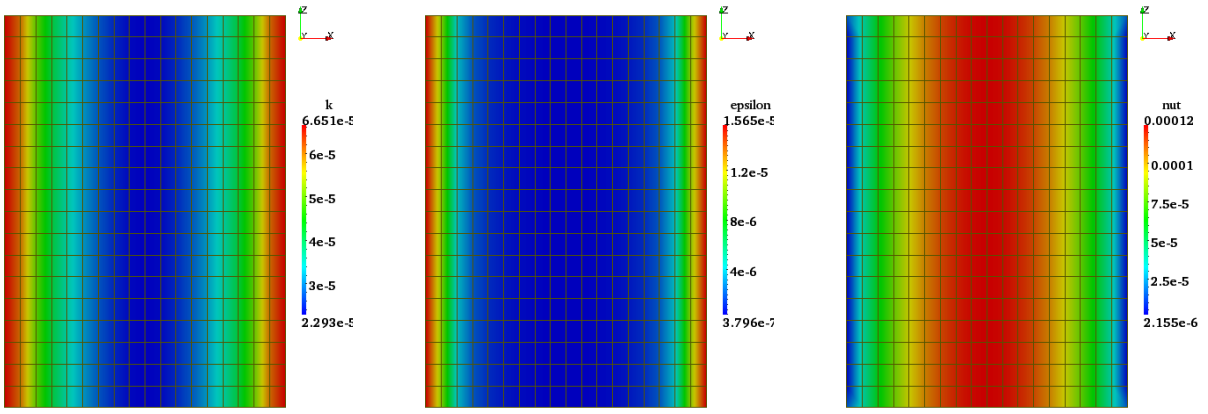
Figure *4.7* shows the kinetic energy k, dissipation rate ϵ and turbulent viscosity ν_t fields as visualized in ParaView on plane ($x \in [0:0.5], y = 0.25, z \in [0:0.7]$). Appended figures *A.12* and *A.19* show the same for case studies 2 and 3.

These fields did not provide any additional insight in terms of addressing study objectives however they do give some indication of flow character which is not apparent from velocity profiles. Figure *4.7 a* quantifies turbulent kinetic energy at the order of 10^{-5} which is not expected to cause severe particle dispersion. All turbulent quantities increase with flow velocity as noted when comparing case studies. This was to be expected as higher turbulent energies are accompanied by higher dissipation rates, the ratio of which is related to the fictitious turbulent viscosity *4.7 c*, and is an indication of

flow resistance brought about by turbulence.

Dissipation rate fields at higher flow velocities figures *A.12 b* and *A.19 b* were heavily contrasted to that in case study 1 figure *4.7 b* i.e. relatively large dissipation at the walls diminishing to field minima within 2 or 3 *cm* whereas case study 1 shows a more gradual reduction in dissipation rate moving away from the walls. Turbulent kinetic energy figure *4.7 a* is greatest at the walls where it is generated, as is the case with dissipation rate figure *4.7 b*, however the dissipation rate is comparatively a lot lower in the bulk flow compared to that at the walls. Since turbulent viscosity is proportional to the ratio of kinetic energy and dissipation rate, turbulent viscosity is kept low at the walls where dissipation is high, indicative of fluid to wall asperity interaction.

Coupled simulation only requires a fluid velocity profile in order to compute momentum exchange terms between the discrete and continuous phases. It should be noted that stochastic Lagrangian eddy models have been formulated, in which particle eddy interaction is accounted for as a coupling force based on these turbulence fields, hence their usefulness should not be underestimated simply on the grounds of their inapplicability in this study. Wall adjacent kinetic energy and viscosity fields may be useful in determining y^+ and y^* in the absence of alternate data.



(a) *k Field* ***(b)*** *ϵ Field* ***(c)*** *μ_t Field*

Fig. (4.7) *Kinetic energy, dissipation rate and turbulent viscosity fields, xz slice plane*

4.1.4 Summary

Findings

Some important findings which can be summarized from flow development results presented in sections *4.1.1* through *4.1.3* include:

i simulated pressure to mean velocity relationships were validated against the Darcy-Weisbach equation

ii the capped profile characteristic in turbulent pipe and duct flow as reported in literature, was

recovered through simulation

iii the log of the wall velocity profile, modelled by wall functions was confirmed. The two y parameters were found to be comparable in magnitude

iv the z velocity flow profile in the bulk region where particles were to be inserted was found to become grid independent around 14 elements, but simulations were run at 18 elements in the primary flow direction

v the velocity profile did not change considerably between simulation times $t = 1800$ and $t = 3000$ s. Pressure gradient remained constant for the duration of simulation

vi turbulent profiles may be useful in extending the model incorporating stochastic Lagrangian models in the coupled framework

Time Statistics

Table *4.3* gives an overview of simulation times required to develop the three flow profiles, for comparative purposes. Time increment was continuously adapted to conform to the Courant number limiter Co_{lim} which was set at 0.95 for each case. A maximum time increment Δt_{max} was specified at 1 s.

It is instructive to note that there seems to be strong correlation between computational time and mesh density, refer to table *4.4*, irrespective of simulation time. The difference between execution time, which is actual CPU time and clock time is that required to write files and perform other extraneous operations. It should also be noted that the centreline flow velocities are approximate values, exact values obtained through simualtion can be found in table *4.2*.

Tbl. (4.3) *CFD time statistics*

[a] $U_z\ \left(\frac{m}{s}\right)$	[b] Sim. Time	[c] Exec. Time	[d] Clock Time	Δt	[e] Δt_{max}	[f] Co_{lim}
0.1	3000	604	779	ATS	1	0.95
0.5	1000	899	911	ATS	1	0.95
1.0	600	2044	2084	ATS	1	0.95

[a] approximate velocity at the duct centreline
[b] simulation time in computational seconds
[c] CPU execution time (s)
[d] wall clock time (s)
[e] Δt_{max} time step limiter
[f] Co_{lim} Courant number limiter
ATS adaptive time stepping

Data Table

Table *4.4* facilitates comparison between the three CFD simulation runs. Values obtained for pressure gradients, y parameters, boundary layer thickness's, mesh densities and wall grading factors are tabulated. Larger flow rates and denser meshes call for smaller wall grading factors to maintain y

parameters in check.

Tbl. (4.4) *CFD data*

[a] U_z $\left(\frac{m}{s}\right)$	∇p $\left(\frac{m}{s^2}\right)$	y^+	y^*	[b] BLT (cm)	[c] (n_x, n_y, n_z)	Grading Factor
0.1	$1.0\,e^{-4}$	64.2	62.0	3.14	(18, 18, 18)	1
0.5	$1.8\,e^{-3}$	93.5	88.0	3.34	(20, 20, 18)	0.2
1.0	$6.5\,e^{-3}$	112.7	102.4	3.29	(26, 26, 18)	0.15

[a] approximate velocity at the duct centreline
[b] Boundary Layer Thickness (BLT)
[c] mesh elements, total mesh elements = $n_x \times n_y \times n_z$
y^+ and y^* values correspond to converged state flows

4.2 CFD-DEM

This section will deal primarily with particulate system response, beginning with a brief discussion on the numerical **Ksl** and **Us** coupling fields, followed by some words on physical validation, considering observable trends noted in published experimental work. The ensuing sub-sections will cover in more detail particle trajectories and acoustic and drag force contributions. Most of the graphical and tabulated data presented pertains to a 6 particle sample specifically selected to cover the size range. A list of all particle sizes and densities may be found in appendix table *B.9*. Sampled particles for which data has been produced are greyed in the appended table. The focus will be on results obtained from case studies 1 and 2.

The coupled simulation was mainly concerned with the particulate phase and is hence run at small time increments over a short simulation period to capture only the particulate phase response to the fluid acoustic system. Simulation times are discussed at the end of this section in *4.2.4* table *4.8*. This limited particle travel eliminates any difficulties with periodic particle mapping and is sufficient to draw conclusions relating to the efficiency of acoustic deflection.

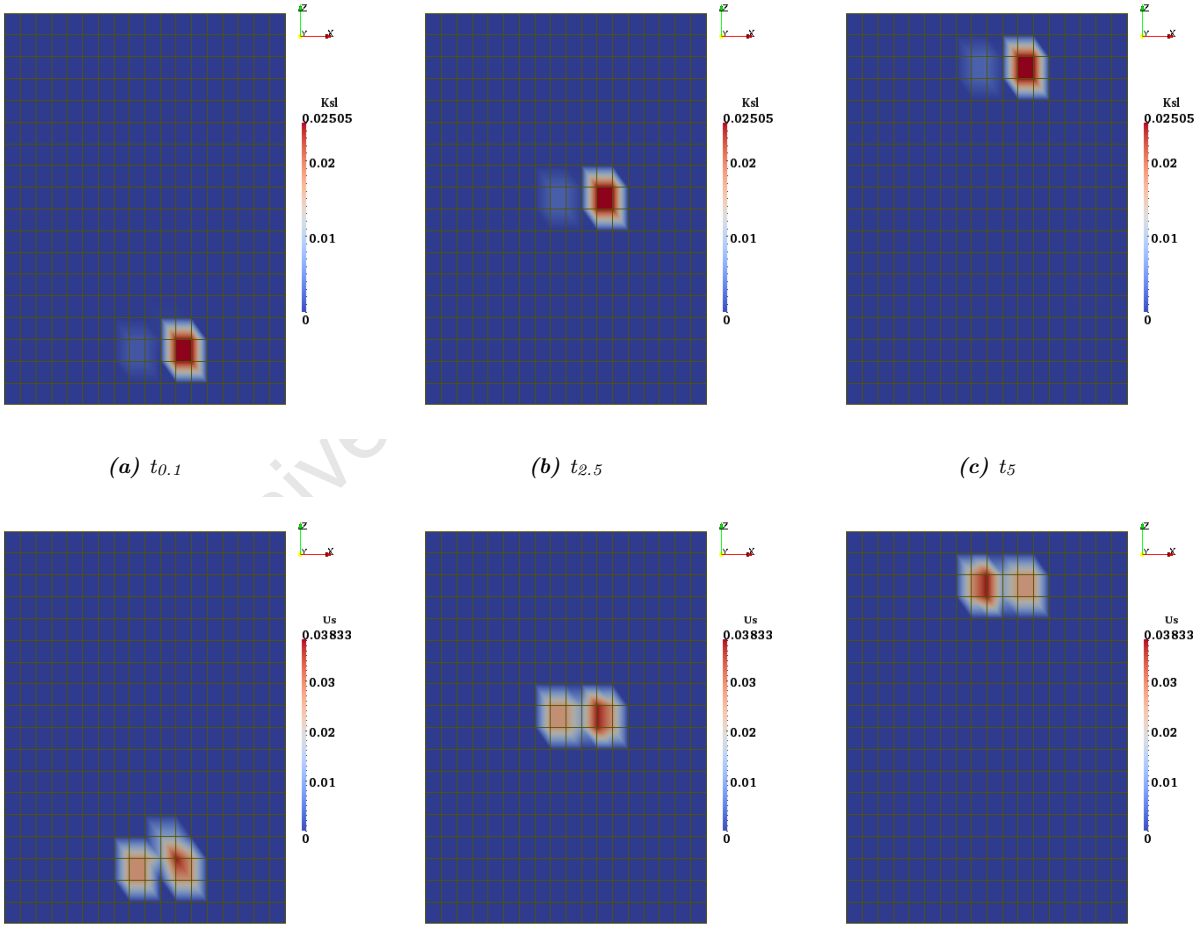
Raw output data from LIGGGHTS simulation, generated by modified dump functionality, detailed in section *3.3.2*, was post-processed in MatLab to produce appended tables *B.10* through *B.15*. Greyed rows designate important information which has been subsequently extracted as necessary for presentation in the following sections. Raw data was used to plot trajectories and forces on selected particles. The functions used to post-process particle related data were designed to handle any number of particles, not limited to those 6 selected.

4.2.1 Validation

This section will present some numerical results based on fields produced during coupled simulation, followed by an overview of pertinent simulation results and trends which have also been noted qualitatively in published literature.

Numerical

Figures *4.8 a* to *c* and *d* to *f* show time evolution of particle continuum velocity **Us** and continuum momentum exchange **Ksl** fields respectively, on plane ($x \in [0:0.5], y = 0.25, z \in [0:0.7]$). Only certain particle contributions are registered on this plane hence the figures are not representative of the system but can give a general indication of numerical progression. These fields are documented on qualitative grounds to illustrate the time evolution of fields generated by the coupling code CFDEM. It should be noted that time is sampled from $t = 0.1\ s$ and not from $t = 0\ s$, as these fields are null at time $t = 0\ s$ because particles have no initial velocity or force acting on them hence no momentum exchange is registered. Because of the small size of the particles and the low solid concentration in the system the **Ksl** field contributed little to momentum exchange from particles to fluid, and hence the fluid remained unaffected by the particulate system. The necessity of using two way coupling was thus brought into question in terms of the additional computation time which was incurred, nevertheless the implementation of two way coupling and the resulting unaffected flow field served as confirmation of the dilute nature of the system. Accuracy was not compromised by implementing two way coupling.



(a) $t_{0.1}$ ***(b)*** $t_{2.5}$ ***(c)*** t_5

(d) $t_{0.1}$ ***(e)*** $t_{2.5}$ ***(f)*** t_5

Fig. (4.8) *Ksl and Us fields on the xz cross sectional plane, f=14794 Hz*

Physical

Considerations regarding physical validation of the model were outlined in section *3.4*. Although the large scale model cannot be validated against real systems due to both a lack of available experimental data and some inherent oversimplified physics, the force model can however be compared against analytic forms, figure *A.5* and previous experimental work by assessing general response trends, which are discussed in sections *4.2.2* and *4.2.3*. Many researchers simply state the effect of the acoustic force as known to either produce agglomeration or clumping [8, 4, 12], or deflect particles to specific planes [11, 12, 10, 37][51] depending on the topic of investigation. Most of the papers reviewed dealing with micro chamber systems use the acoustic force of equation *2.1.6* to produce numerical models which are then validated using experimental rigs. Agglomeration is more difficult to model numerically hence pure experimental type studies are more common when investigating larger systems. Hawkes *et al* [11, 12] and Whitworth and Coakley [41] amidst others note the observed striated particle patterns which emerge in the presence of an acoustic field. These are qualitatively reproduced by our simulation, see section *4.2.2*. They also note the influence of frequency on striation density and filtration efficiency i.e. higher frequencies produce more closely spaced particle lines and more agglomeration, the former also being reproduced by our simulation and consistent with the acoustic equation on which the model was based. The secondary acoustic force of equation *2.1.11*, responsible for agglomeration, is dependent mostly on the same factors as the primary acoustic force i.e. driving frequency, driving amplitude, particle size and inter-particle spacing. Most papers acknowledge the fact that larger particles are better deflected [15], agglomerated and form more stable striation bands [11], attributed to larger acoustic forces also reproduced qualitatively by our simulation, see section *4.2.3*. Most published work involves some carrier fluid and a drag force, the flow rate which is adjusted in relation to the acoustic field intensity to produce the desired particulate phase response whether it be deflection or agglomeration. Hawkes [12] notes decreased filtration efficiency under higher flow rates, also qualitatively reproduced, see section *4.2.2*, assuming that filtration efficiency is related to the strength of the acoustic forces in relation to active transducer length and residence time.

4.2.2 Particle Trajectories

This section will present trajectory results obtained from simulation. All relevant trajectory data pertaining to the 6 particle sample have been condensed from appendix tables into table *4.5*. The selected particles cover the size range of the applied distribution, so we can investigate the effect particle size has on deflection. All tabulated data differs with respect to three primary factors namely driving frequency, flow rate and particle size.

Particle deflection efficiency was measured as the ability of an acoustic system to drive a given particle to within a certain tolerance distance of its nearest pressure node. The tolerance was arbitrarily selected at 1.5 mm but would otherwise be determined through design considerations. Nodal spacing at a driving frequency of 14794 Hz was around 50 mm and at 26629 Hz around 28 mm, refer to table *B.5*. This placed the tolerance specification at 3 and 5 percent of the respective nodal separation distances. Particles which have been randomly seeded are more likely on average to be nearer these

pressure nodes under higher driving frequencies than under lower ones resulting in smaller deflections.

If deflection to within the predefined tolerance of the nearest pressure node had been achieved an entry is recorded in table *4.5*, and similarly in appended tables *B.10* through *B.15*, otherwise the particle was not deflected to within the 1.5 mm tolerance and is not registered, denoted by a hash line.

Table *4.5* entries reveal four important points **i** to **iv**.

i whether or not a particle had been sufficiently deflected to within 1.5 mm of its nearest pressure node, denoted by registry

ii lateral deflection δx of the particle in aligning with its nearest pressure node. It should be noted that the δ is used to denote particle displacement from initial position and should not be confused with domain coordinates

iii transducer length required to deflect the particle to within 1.5 mm of its nearest pressure node, corresponding to the vertical particle travel distance δz

iv residence time δt required to deflect the particle to within the specified 1.5 mm distance of its nearest pressure node

v (example) a particle of radius 5.1 micron, with identifier number 6, was deflected 1.03 cm in the x direction under a flow rate of 0.1 ms^{-1} and driving frequency 26629 Hz, before it was registered within the 1.5 mm tolerance distance of its nearest pressure node. The time taken for particle 6 to attain this x coordinate was 0.53 s, and during this time the particle was registered to have travelled 5.33 cm in the z direction.

Residence time in the context of this study refers to that time period, irrespective of particle travel distance or its velocity, for which a particle must be subject to an acoustic field to attain sufficient deflection based on a tolerance distance from its adjacent nodal plane. It is important to consider that residence time varies with frequency but remains independent of flow rate.

Tbl. (4.5) *Trajectory data*

$U(\frac{m}{s})$	$f(Hz)$	p_{id}	13	6	5	4	19	2
		$r\,(\mu m)$	5.0	5.1	9.2	17.1	18.5	25.7
0.1	14794	(δx, δz, δt)	(-1.13, 41.94, 4.11)	–	(0.17, 3.98, 0.40)	(-1.25, 4.68, 0.46)	(0.78, 3.06, 0.31)	(-1.15, 2.28, 0.23)
	26629	(δx, δz, δt)	(0.24, 1.83, 0.18)	(1.03, 5.33, 0.53)	(-0.65, 1.19, 0.12)	(-0.17, 0.20, 0.02)	(-1.15, 0.78, 0.08)	(0.88, 0.39, 0.04)
0.5	14794	(δx, δz, δt)	–	–	(0.17, 19.44, 0.40)	(-1.24, 22.58, 0.46)	(0.78, 14.76, 0.30)	(-1.15, 10.95, 0.23)
	26629	(δx, δz, δt)	(0.24, 8.69, 0.17)	(1.03, 25.93, 0.53)	(-0.65, 5.83, 0.12)	(-0.14, 0.74, 0.01)	(-1.19, 4.08, 0.09)	(0.84, 1.69, 0.04)
1.0	14794	(δx, δz, δt)	–	–	(0.17, 38.43, 0.40)	(-1.24, 45.04, 0.45)	(0.77, 29.50, 0.30)	(-1.14, 21.79, 0.22)
	26629	(δx, δz, δt)	(0.23, 17.07, 0.17)	–	(-0.65, 11.45, 0.12)	(-0.13, 1.39, 0.01)	(-1.17, 8.00, 0.08)	(0.80, 3.13, 0.03)

Table notes

δx x deflection distance (cm)

δz particle travel length corresponding to a lateral deflection to within 1.5 mm of its nearest node (cm)

δt time period over which a particle is subject to an acoustic field corresponding to its lateral deflection to within 1.5 mm of its nearest node (s)

Final particle positions may be found in appended tables, denoted by subscript *f*. Final positions may be referred to in determining whether a particle was retained at the nodal position for the duration of its passage after having been registered initially within nodal tolerance distance. It was noted that particles are in fact retained in all cases at their respective pressure nodes once driven to them initially. Additional convergence to exact nodal coordinates was also noted, and was to be expected in the absence of disruptive forces. Exact nodal coordinates are tabulated in *B.5*. Particles were indeed deflected to pressure nodes, and not anti-nodes, ascertained by comparing final x_f and nodal x_n coordinate entries with those tabulated based on analytic forms. This held true in all cases which had been investigated, by virtue of the acoustic contrast factor which is a function of fluid to particle density and compressibility ratios. It should also be noted that particles retained straight trajectories in the yz plane unaffected by the acoustic force, refer to table *B.10* rows 8-10.

By analysing particle displacement data we ascertained that residence time was inversely proportional to particle size, but also related to the initial particle node separation distance as was noted when comparing particles 5 and 4, table *4.5* under a driving frequency of 14794 Hz and particles 13 and 6 under a driving frequency of 26629 Hz. Particle 5 was smaller than particle 4 but registered shorter residence times and vertical travel lengths by virtue of its immediate near nodal alignment. It is explicitly clear when considering particles 13 and 6 which are of similar dimension, but have markedly different residence times, that particle node separation distance contributes to deflection response.

The required transducer length increased not only with residence time but also with flow velocity. Because of its relation to residence time transducer length was inversely proportional to particle size and directly proportional to flow velocity. Only those particles with residence times less than that of the respective simulation were registered as being adequately deflected. Particle 13 was not registered in case studies 2 and 3 at the lower driving frequency $f = 14794\ Hz$ because simulations did not run longer than the required residence time of 4.11 s. The longer the residence time the longer the required active transducer length.

At a flow rate of 1 ms^{-1}, driving frequency 14794 Hz, a transducer length of around 45 cm was required to deflect particle 4, 17.1 micron radius sufficiently, which corresponded to a residence time of 0.45 s. The simulation had been run for 0.5 s hence capturing the deflection near the end of the run. Higher driving frequencies are associated with lower residence times resulting in more particles being deflected to within nodal tolerance distance over their respective travel lengths.

Figures *4.9* and *4.10* show particle trajectories for the 6 sampled particles at various flow velocities. Trajectories taken under a driving frequency of 14794 Hz are shown in figure *4.9* and those taken under a driving frequency of 26629 Hz are shown in figure *4.10*. Each particle is assigned a colour and flow velocities are differentiated by line style. The figures are more illustrative of deflection response however tabulated data should be referred to in cases where trajectory plots are unclear. By examining corresponding entries in table *4.5* the predefined tolerance of 1.5 mm predicts particle 4 under a driving frequency of 14794 Hz and flow velocity 0.5 ms^{-1} to travel 22.58 cm at registry, in reasonable agreement with the trajectory plot for the same particle showing that the particle attains a graphically resolved nodal coordinate at around 35 cm, bearing in mind its initial position at around 10 cm. Similar consistency between tabulated and graphical data may be noted for all sampled parti-

cles at either frequency. This lead us to conclude that the tolerance could be adjusted in accordance with graphical data, however the deflection does not have to be total for design application and hence the tolerance can be adjusted in accordance with the level of precision required for the system. It should also be noted that convergence rate decreases as a particle nears a pressure node which has further implications with regard to selecting a suitable tolerance level. This decreasing rate of convergence may be attributed to the decreasing acoustic force as a particle nears its respective nodal plane coordinate. The tolerance may be easily adjusted to produce revised data however the need was not warranted as tabulated and graphical data were evidently complimentary.

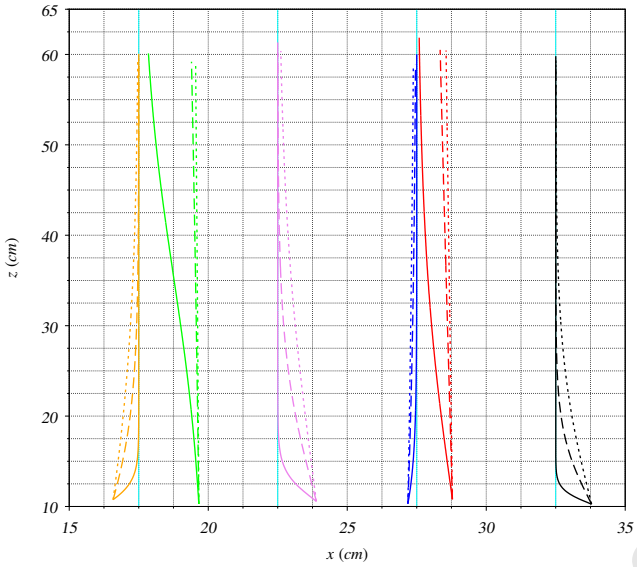


Fig. (4.9) *Particle trajectories at various flow velocities, f=14794 Hz*

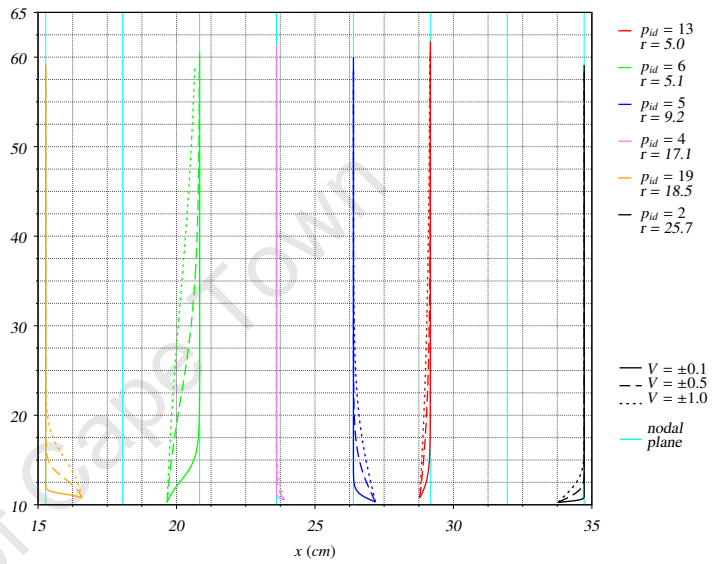


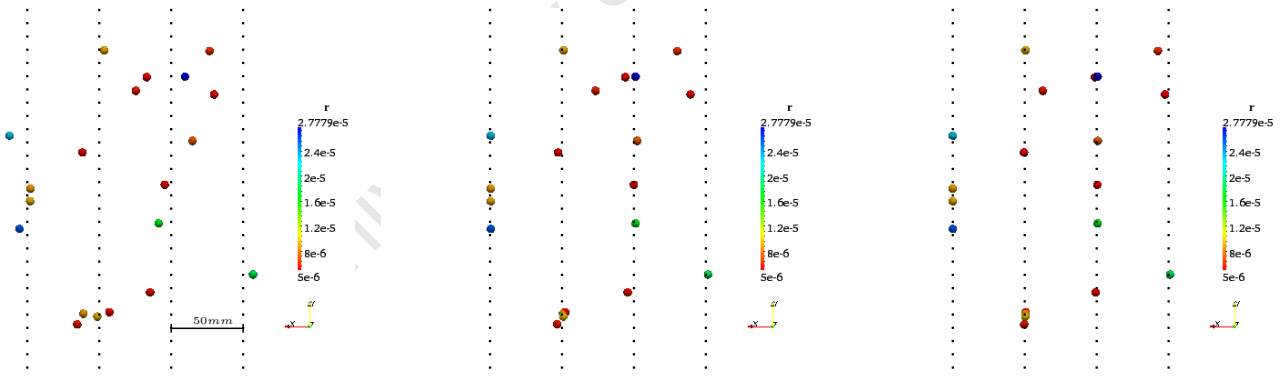
Fig. (4.10) *Particle trajectories at various flow velocities, f=26629 Hz*

A critical height level at which most if not all particles which could be adequately deflected, were deflected, may be determined by viewing figures *4.9* and *4.10* or picking out maximum vertical travel lengths from tabulated data for a set of particles and parameters. This height progressively increases with flow rate. For example under a driving frequency of 14794 Hz, flow rate 0.1 ms^{-1} this height could be around 15 cm, corresponding to a 5 cm transducer length. This level however excludes the two 5 micron particles which would require an active acoustic height in excess of 42 cm. Similar analyses were undertaken for the remaining cases and it was concluded that for the lower driving frequency, considering only particles larger than 9 micron in radius, transducer lengths ranging from 5 to 22 to 45 cm would be required for the various flow velocities. For the higher driving frequency, considering particles larger than 9 micron, transducer lengths ranging from 1 to 5 to 12 cm would be required, a considerable improvement. In addition certain smaller particles were also adequately deflected at 26629 Hz. At this frequency and a 1 ms^{-1} carrier velocity particle 13, 5 micron was adequately deflected over a 17 cm travel distance. Graphically particle 6 which remained unregistered at the highest flow rate, seemed nearer its nodal plane at simulation end time. It should be noted however that these critical levels were obtained through investigation of only those 6 sampled particle trajectories covering the critical portion of the size range. Additional trajectories could offset the values documented. Considerably improved deflection response was achieved at higher frequencies

as evident by comparing figures *4.9* and *4.10*. The improved deflection may be overestimated if not considered in relation to particle node separation distance which is reduced on average under these higher frequencies.

Visuals, figures *4.11* to *4.14* show qualitatively particle system behaviour for case study 1. Figures *4.15* to *4.18* pertain to case study 2. Those pertaining to case study 3 are appended. The lower resonant frequency around 14794 Hz was selected as the nodes would be spaced farther apart for a more visually pronounced effect, however even at this relatively low frequency practical separation distances were not achievable, hence the simulation runs served more as conceptual proofs than implementable system models. Higher driving frequencies produced denser node fields in linear planes as illustrated in figure *A.5*, and hence denser particle striations were expected. Typically higher frequencies are used to produce agglomeration by reducing inter-particle separation distance through planar alignment, and thereby facilitating the action of secondary acoustic and other inter-particle forces which act at short range to produce the desired clumping.

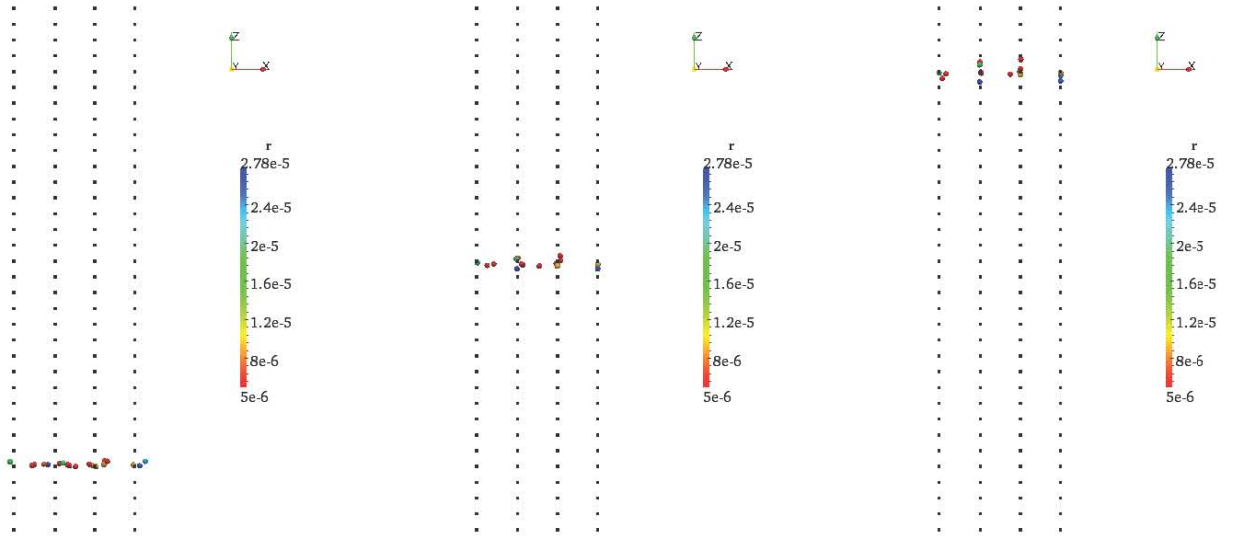
The following visuals were obtained by zooming into particle concentrated areas capturing a domain segment bounding a number of nodal planes. Nodal lines were superimposed on the figures to show particle alignment with nodal planes. Nodal planes are regularly spaced, dependent on driving frequency as documented in table *B.5*. Time snapshots are documented at three times, initial time, some midway point and end time. Despite the fact that particles are randomly seeded into the specified region, a consistent integer seed produces the same random number propagation sequence and hence the particles are distributed as before. The particles were colour coded in terms of their radial size.



(a) t_0, f=14794 **(b) $t_{2.5}$, f=14794** **(c) t_5, f=14794**

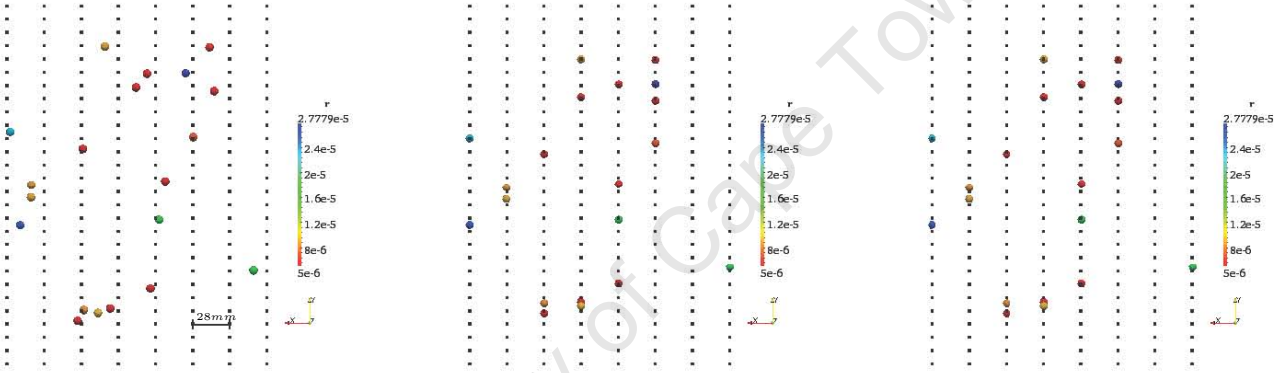
Fig. (4.11) *Case study 1, $U_z = 0.1\ ms^{-1}$, particle visualization xy plane, f=14794 Hz*

Smaller particles are seen to be less affected by the acoustic force and are last to align with nodal planes. On the other hand slightly larger particles responded much quicker to the acoustic field. These findings were consistent with previous plot and tabulated data. The advantage of this type of visual representation is that the entire system of particles may be viewed and not only those sampled few. All cases exhibit similar behaviour i.e. the system at time zero is disordered and becomes more ordered as the simulation progresses. Under higher driving frequencies as in figures *4.13* and *4.14* order is attained much sooner than under lower driving frequencies.



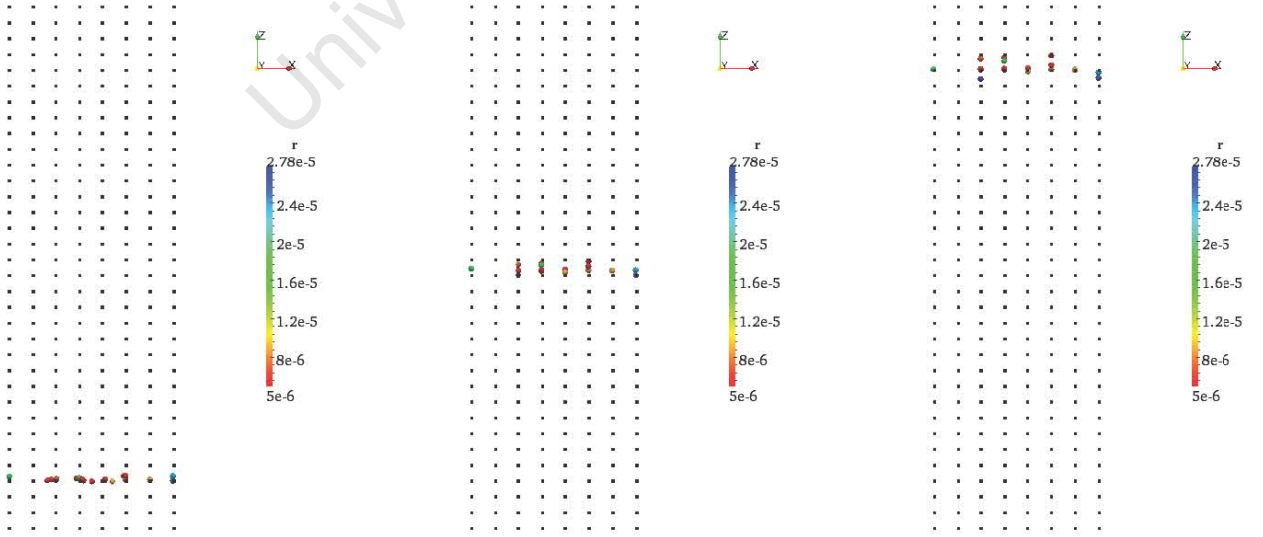
(a) t_0, f=14794 **(b) $t_{2.5}$, f=14794** **(c) t_5, f=14794**

Fig. (4.12) *Case study 1, $U_z = 0.1\ ms^{-1}$, particle visualization xz plane, f=14794 Hz*



(a) t_0, f=26629 **(b) $t_{2.5}$, f=26629** **(c) t_5, f=26629**

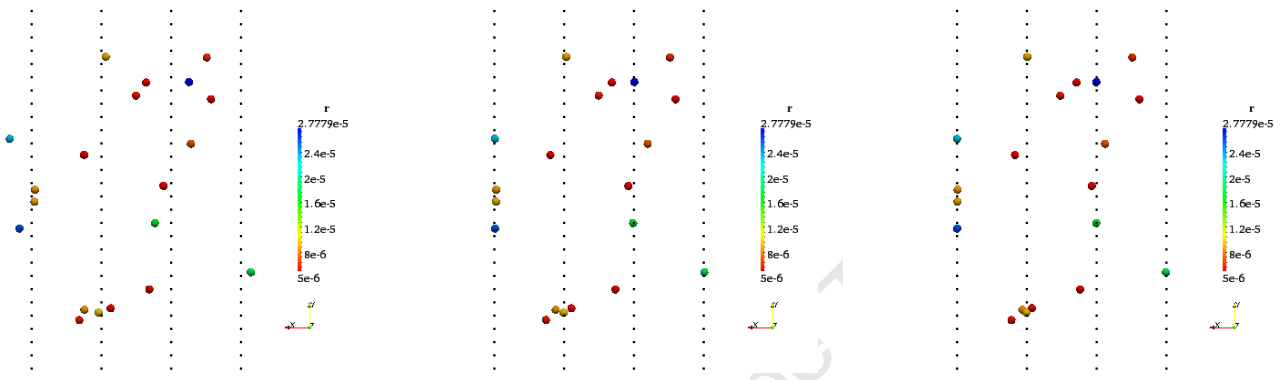
Fig. (4.13) *Case study 1, $U_z = 0.1\ ms^{-1}$, particle visualization xy plane, f=26629 Hz*



(a) t_0, f=26629 **(b) $t_{2.5}$, f=26629** **(c) t_5, f=26629**

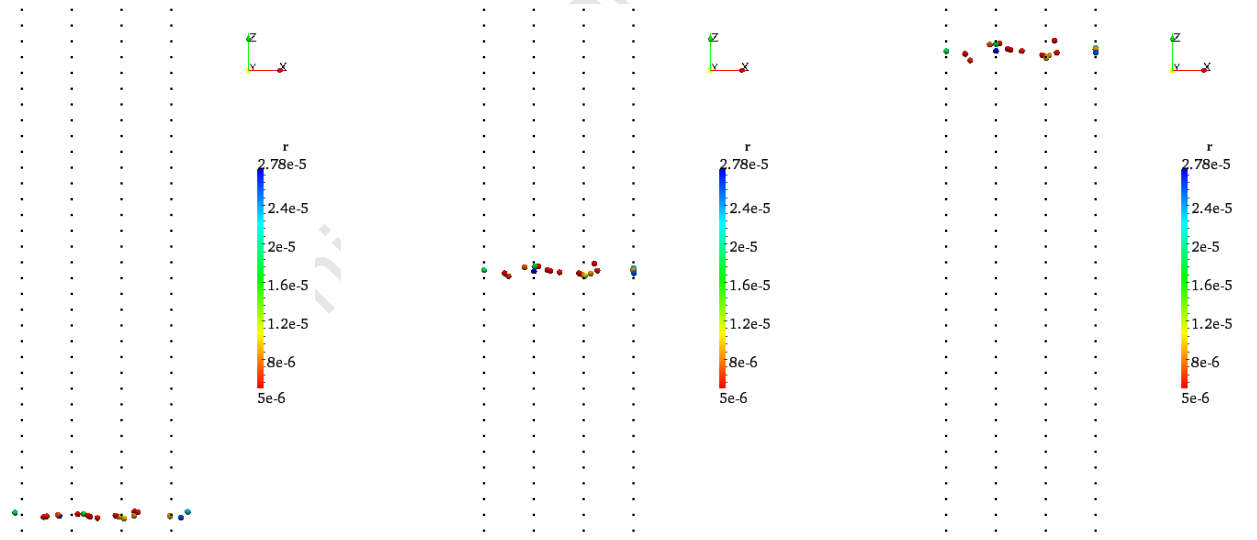
Fig. (4.14) *Case study 1, $U_z = 0.1\ ms^{-1}$, particle visualization xz plane, f=26629 Hz*

Under higher flow rates and lower driving frequencies such as in figures *4.15* and *4.16* the system of particles is generally more disordered over the given travel distance as the time exposure to the acoustic force is less. Smaller particles in faster streams were less responsive at lower driving frequencies, hence larger frequencies were required to achieve the desired particle deflection. High frequency low flow rate systems produce more aligned configurations consistent with previous findings. Particles at the blue end of the size spectrum corresponding to larger radii align at nodal planes by the second frame at mid point simulation time in all cases. Certain smaller particles near the red end of the size spectrum were found unaligned even at simulation end times in particular cases.



(a) t_0, f=14794 **(b) $t_{0.5}$, f=14794** **(c) t_1, f=14794**

Fig. (4.15) *Case study 2, $U_z = 0.5\ ms^{-1}$, particle visualization xy plane, f=14794 Hz*

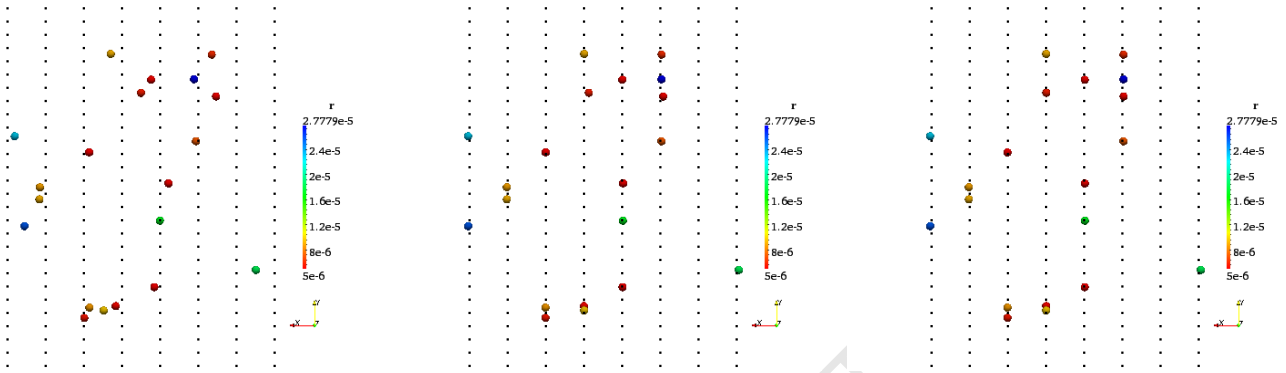


(a) t_0, f=14794 **(b) $t_{0.5}$, f=14794** **(c) t_1, f=14794**

Fig. (4.16) *Case study 2, $U_z = 0.5\ ms^{-1}$, particle visualization xz plane, f=14794 Hz*

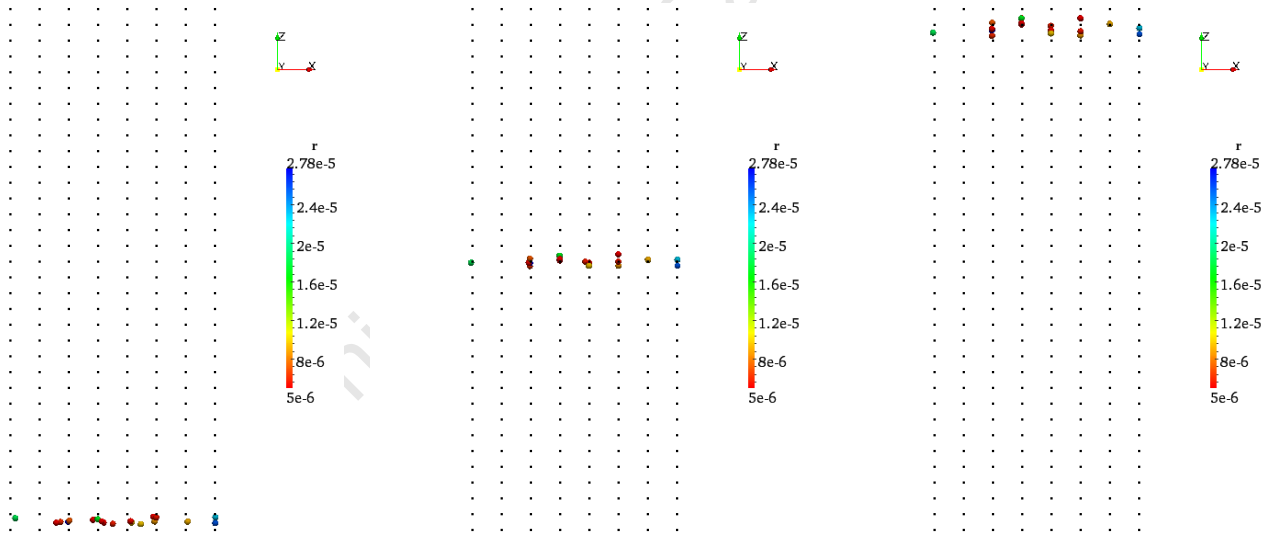
By studying the xy plane visuals at the respective final time frames, we can ascertain the numbers of particles which were not adequately deflected at various flow rates and driving frequencies. Under a driving frequency of 14794 Hz at the low flow rate 0.1 ms^{-1} 2 of 20 particles remained unaligned with pressure nodes, at 0.5 ms^{-1} 8 of 20 particles remained unaligned and at 1.0 ms^{-1} 11 of 20 remained

unaligned. Under the higher driving frequency 26629 Hz all particles were graphically aligned at simulation end times except for a single 5 micron particle at a flow rate of 1.0 ms^{-1}. This was indeed particle 6 as tabulated. The improved alignment response at higher driving frequency was attributed to an increased acoustic force in conjunction with more closely spaced nodes. Generally smaller particles sized between 5 and 10 micron, when compared to their neighbouring particles responded in a delayed fashion to the applied acoustic field under the investigated range of parameters.



(a) t_0, f=26629 **(b) $t_{0.5}$, f=26629** **(c) t_1, f=26629**

Fig. (4.17) *Case study 2, $U_z = 0.5\ ms^{-1}$, particle visualization xy plane, f=26629 Hz*



(a) t_0, f=26629 **(b) $t_{0.5}$, f=26629** **(c) t_1, f=26629**

Fig. (4.18) *Case study 2, $U_z = 0.5\ ms^{-1}$, particle visualization xz plane, f=26629 Hz*

4.2.3 Forces

This section will deal separately with each of the two most significant forces, namely the Primary Acoustic force and the drag force. Buoyancy and gravity seemingly played a lesser role and are hence neglected from the core discussion. The two forces are treated individually because they act in perpendicular planes and are thus decoupled.

Primary Acoustic Force

In contrast to particle trajectory analysis where tabulated data was definitive, force data is better presented using graphical means. Plots are supplemented by condensed tabulated data extracted from appended tables for each case study.

Figures *4.19* and *4.20* show the time evolving x force component magnitude acting on selected particles, under the two driving frequencies, differentiated by line style. The colours previously assigned to the different particles are retained. The two figures are essentially the same except for the time length represented. Acoustic forces do vary with driving frequency but not with flow velocity reiterating the fact that the forces were decoupled. The plots illustrate the dependence of acoustic force on particle size i.e. larger forces corresponding to larger particles. Larger initial forces are noted under the higher driving frequency for all particle sizes, however these quickly plummet and become somewhat distorted, attributed to numerical and damping phenomena. Some general trends are preserved in terms of relative force magnitudes but these are not clear. It should be noted that the figures do not take the sign of force into account.

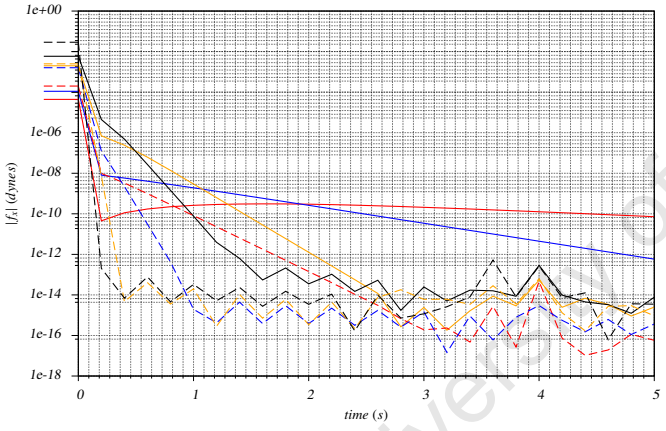


Fig. (4.19) *Primary acoustic force at two driving frequencies t ∈ [0:5]*

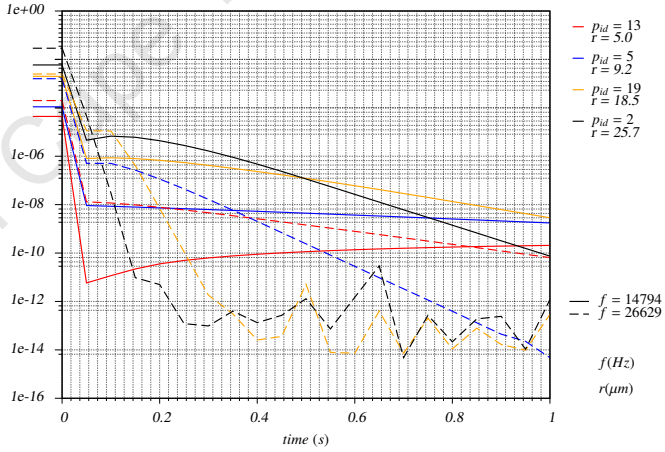


Fig. (4.20) *Primary acoustic force at two driving frequencies t ∈ [0:1]*

The figures are accompanied by table *4.6* which contains initial and final time force magnitudes denoted by $|f_x|_{in}$ and $|f_x|_f$, as well as bracketing forces registered for the duration of simulation denoted by fx_{hi} and fx_{lo}, which take sign into account. Forces registered at initial and final times give some indication of convergence in the x direction.

At the onset of simulation particles are located at their farthest points from respective pressure nodes and thus subject to largest acoustic forces in the x direction. This is evident from the figures and consistent with corresponding data entries in table *4.6* at initial time. As these particles move toward nodal planes these acoustic components diminish by virtue of a reducing particle node separation distance. Particles subsequently experience an opposing force to the initial acoustic drive resulting in damped response. The tabulated forces subscripted with *hi* and *lo* correspond to the initial driving force directed toward a respective nodal plane and the correcting force, the 2nd peak which is brought

about by viscous damping acting against the inertial velocity of a given particle, which overshoots initially, hence for each particle there will be one positive and one negative force. The initial force maxima diminish by at least two orders of magnitude within the first damp cycle, however the force reduction can be as large as several orders. Initially seeded particles have zero x velocity components and are only acted on by acoustic forces explaining the large initial peaks which are registered. Subsequently particles attain non zero velocities in the x direction which invoke damping forces in opposition of the acoustic drive which results in a reduction of net force on a given particle. Larger particles are more heavily damped resulting in larger force reductions at the initial stages of simulation. This may be attributed to stronger viscous forces related to particle size and inertia.

Tbl. (4.6) *Acoustic force data*

p_{id}	$r(\mu m)$	f = 14794Hz				f = 26629Hz			
		$\|fx\|_{in}$	$\|fx\|_f$	fx_{hi}	fx_{lo}	$\|fx\|_{in}$	$\|fx\|_f$	fx_{hi}	fx_{lo}
13	5.0	4.4e-05	2.1e-10	2.1e-10	-4.4e-05	2.0e-04	6.5e-11	2.0e-04	-1.3e-08
6	5.1	1.9e-05	2.9e-10	2.2e-10	-1.9e-05	1.3e-04	6.1e-10	1.3e-04	-1.5e-08
5	9.2	1.1e-04	1.7e-09	1.1e-04	-9.8e-09	1.6e-03	4.8e-15	5.9e-07	-1.6e-03
4	17.1	1.7e-03	1.0e-08	6.0e-07	-1.7e-03	6.0e-03	4.2e-14	2.0e-05	-6.0e-03
19	18.5	2.0e-03	2.9e-09	2.0e-03	-8.7e-07	2.5e-03	2.9e-13	2.9e-05	-2.5e-03
2	25.7	5.9e-03	7.5e-11	6.8e-06	-5.9e-03	2.9e-02	1.2e-12	2.9e-02	-1.9e-04

Table Notes

$|fx|_{in}$ *initial force magnitude recorded (dynes)*

$|fx|_f$ *final force magnitude recorded (dynes)*

$fx_{hi,lo}$ *force bracket for the duration of simulation (dynes)*

results are valid for all flow velocities

Once the second force peak attributed to the damping force is registered the force falls even further. Particles continue their convergence paths toward nodal plane equilibrium positions, the forces and displacements tending toward machine precision as the simulation progresses, since there are no disrupting forces. Machine precision on a 64bit machine is around $1e^{-16}$ which the computer registers as zero. Once a particle has passed the initial transient cycle convergence is much slower as relatively small forces are in effect. In the absence of viscous drag forces the particles would inevitably oscillate about nodal coordinates with constant energy intensity.

Acoustic force is a function of particle node separation distance hence varies with frequency and random particle seed. By comparing acoustic force magnitudes in table *4.6* it was concluded that stepping the driving frequency from 14794 Hz to 26629 Hz resulted in a general one order of magnitude increase in initial acoustic force irrespective of particle node separation distance. An order of magnitude increase in force was noted in 4 of 6 sampled particles, and an increase in force within the same order of magnitude noted in the remaining 2 particles. Due to random proximities of particles and their adjacent nodal planes, peak forces were seldom realized. It was subsequently concluded that the acoustic force was highly dependent on frequency, but also on relative particle node separation distance which reduced the force increase for certain particles. These observations have been noted in

literature and are indicative of trends predicted by the governing equation form.

Density variation within the limited range was found to contribute negligibly toward acoustic force magnitude as all force trends were retained irrespective of density. Particle density is not irrelevant however, its contribution to the acoustic force equation was not in terms of force magnitude but rather in defining whether particles are forced to nodes or anti-nodes as mentioned previously. The limits of the distribution do not fall near critical density at which transition occurs, hence the mean density 2500 kgm^{-3} may be assumed for all particles.

Drag force

By considering velocity entries, row 17 of table *B.10* and similar entries for the other cases, it was clear that particles attained near flow velocities almost instantaneously i.e. within the first write interval. The velocities attained were assumed to be terminal for each particle in simulation. The instantaneous equilibration lead us to conclude that the time required to accelerate particles to near flow velocity was negligible in terms of offsetting deflections. Had particles been more resistant to acceleration the required transducer lengths might have been underestimated because particles would have been travelling at slower rates compared to the surrounding flow over some initial transient period. The inclusion of eddy interaction may also affect system dynamics because particles may experience intermittent stalling and acceleration, thereby facilitating longer or shorter acoustic interaction times. The small velocity differences between particles, which may be noted in tabulated data, were attributed to the distribution of particles within the flow profile.

Drag force is similar to acoustic force in terms of graphical treatment and tabulation, however the z component of force acting on a given particle is plotted for various flow velocities, differentiated by line style, and not for the two driving frequencies. Figures *4.21* and *4.22* are essentially the same with different time interval. The data was plotted for a driving frequency of 14794 Hz.

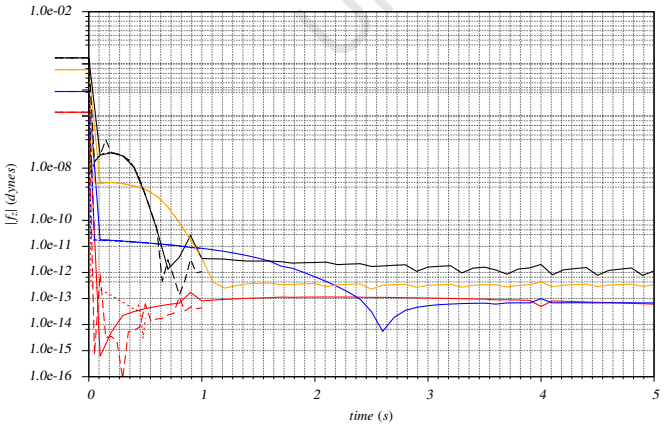


Fig. (4.21) *Drag force at various flow rates, f=14794 Hz t ∈ [0:5]*

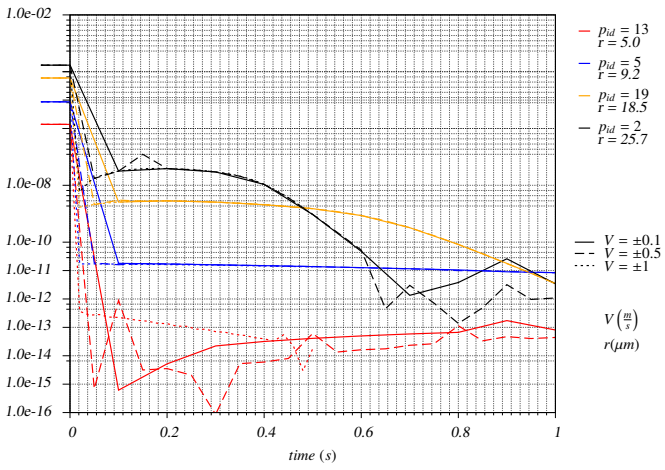


Fig. (4.22) *Drag force at various flow rates, f=14794 Hz t ∈ [0:1]*

Figures *4.23* and *4.24* are presented to show that force time plots obtained under the higher driving frequency are somewhat different to those obtained under the lower driving frequency although both sets fall within the same order of magnitude range i.e. between 10^{-11} and 10^{-15}. The discrepancies were attributed to different lateral particle positioning brought about by the acoustic force under the different frequencies. The accompanying table *4.7* shows a similar array of sampled forces to that detailed for table *4.6* in the previous section, except differentiating data sets on flow velocity instead of frequency. It should be noted that this table corresponds to data extracted only at the lower driving frequency. Data for the higher driving frequency is expected to be similar in terms of magnitudes and trends.

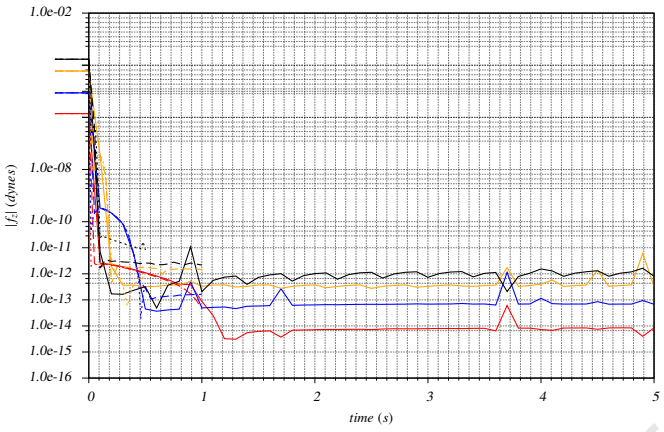


Fig. (4.23) *Drag force at various flow rates, f=26629 Hz t ∈ [0:5]*

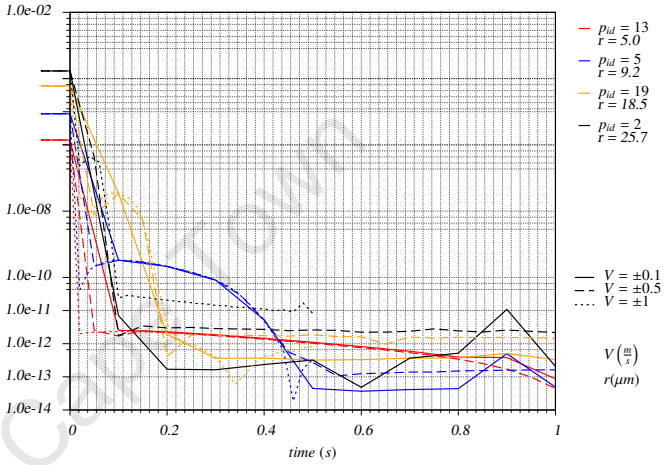


Fig. (4.24) *Drag force at various flow rates, f=26629 Hz t ∈ [0:1]*

All figures show there to be minimal variation in z force for the various flow rates. Initial drag forces were found to respect particle size i.e. larger initial drag forces $|fz|_{in}$ were registered for larger particles, refer to table *4.7* but did not vary with flow velocity which was an unexpected result. By considering tables *4.6* and *4.7* it was noted that initial acoustic forces were 1 and 2 orders of magnitude greater than initial drag forces under a driving frequency of 14794 Hz and 2 and 3 orders of magnitude greater under a driving frequency of 26629 Hz, for the sampled particle set.

A similar damping effect to the one previously explained for the acoustic force was noted here also, refer to fz_{hi} values in table *4.7* which were found to increase in general with particle size. Drag forces peak initially because relative fluid particle velocity in the z direction is at a maximum. Particles attain a near fluid velocity almost instantaneously resulting in a largely diminished drag force reducing by several orders of magnitude depending on particle size and position in the flow field.

Tbl. (4.7) *Drag force data, f=14794 Hz*

p_{id}	$V = \pm 0.1\left(\frac{m}{s}\right)$				$V = \pm 0.5\left(\frac{m}{s}\right)$				$V = \pm 1.0\left(\frac{m}{s}\right)$			
	$\lvert fz\rvert_{in}$	$\lvert fz\rvert_f$	fz_{hi}	fz_{lo}	$\lvert fz\rvert_{in}$	$\lvert fz\rvert_f$	fz_{hi}	fz_{lo}	$\lvert fz\rvert_{in}$	$\lvert fz\rvert_f$	fz_{hi}	fz_{lo}
13	1.4e-06	6.1e-14	1.3e-13	-1.4e-06	1.4e-06	4.4e-14	1.7e-11	-1.4e-06	1.4e-06	1.8e-14	1.1e-10	-1.4e-06
6	1.5e-06	1.4e-13	8.2e-13	-1.5e-06	1.5e-06	1.6e-13	6.3e-12	-1.5e-06	1.5e-06	1.7e-13	8.2e-10	-1.5e-06
5	8.7e-06	6.8e-14	5.1e-12	-8.7e-06	8.7e-06	8.3e-12	-8.1e-12	-8.7e-06	8.7e-06	1.3e-11	2.7e-10	-8.7e-06
4	5.2e-05	2.8e-13	3.1e-10	-5.2e-05	5.2e-05	6.0e-11	3.8e-10	-5.2e-05	5.2e-05	1.6e-09	2.0e-09	-5.2e-05
19	6.0e-05	3.3e-13	1.1e-10	-6.0e-05	6.0e-05	3.4e-12	-3.4e-12	-6.0e-05	6.0e-05	1.5e-09	3.0e-09	-6.0e-05
2	1.7e-04	1.1e-12	3.1e-10	-1.7e-04	1.7e-04	1.1e-12	1.2e-07	-1.7e-04	1.7e-04	9.4e-10	6.7e-06	-1.7e-04

Table notes

$\lvert fz\rvert_{in}$ *initial force magnitude recorded (dynes)*

$\lvert fz\rvert_f$ *final force magnitude recorded (dynes)*

$fz_{hi,lo}$ *force bracket for the duration of simulation (dynes)*

results are valid for either frequency

Drag force is a function of instantaneous relative particle to fluid velocity also known as slip velocity which was expected to increase at higher flow rates. Drag force decreases considerably as the particle approaches fluid velocity by virtue of the decreasing relative fluid particle velocity component. As a particle is accelerated in the flow, slip velocity diminishes and results in a constant drive at equilibrium where the slip velocity is small and relatively invariant. At this stage the system flow velocity would have no effect on the drag force experienced by a suspended particle and force decoupling could be explained. The individual effects of drag, gravity and buoyancy could not be isolated by simply analysing tabulated or plot data. Gravity and buoyancy forces act in the z direction and are also dependent to a large extent on particle size but also on particle density and relative fluid particle density. Their contribution to the z force component remained undistinguished in plot and tabulated data leading us to conclude that their effect was negligible in comparison to that of drag force, possibly attributed to small particle sizes.

4.2.4 Summary

Findings

A few important points, regarding the coupled simulation, sections *4.2.2* and *4.2.3* may be summarized:

i the expected behaviour was simulated. Particulate response exhibited qualitative sensitivity to most acoustic force function dependencies including frequency, particle size and particle node separation distance. It should be noted that driving amplitude was not varied in this study, hence its contribution to the acoustic force was not determined

ii larger particles, higher driving frequencies and lower flow rates produce best deflection results. Smaller particles are found to respond poorly under lower driving frequencies

iii the acoustic force was found generally to be between 1 and 3 orders of magnitude larger than

the drag force for the parameters investigated, based on initial recorded magnitudes

iv acoustic and drag forces were found to increase with particle size, the rate of which is expected to conform to the relevant governing equations i.e. acoustic force equation *2.1.6* which shows a cubic dependence on particle size and the drag force equation *2.2.22* which shows a square dependence on particle size.

v acoustic and drag forces were somewhat decoupled as they act in perpendicular planes, however lateral particle position governed by the acoustic force did affect the z force and velocity components of any given particle, because the flow profile was spatially variant in the cross sectional plane. Since the flow profile was capped, small lateral displacements away from any wall gradients did not produce appreciable changes in these force and velocity components

vi initial acoustic and drag force peaks were attributed to the first acoustic-particle and fluid-particle interaction computations in the absence of resistive damping. The sudden coupling results in large initial transients followed by gradual convergence response

vii damping forces act in opposition to both acoustic and drag forces resulting in orders of magnitude force reductions at the early stages of simulation

Time Statistics

Table *4.8* assimilates CFDEM simulation time statistics for all 6 scenarios, each designated by code, refer to table note a. Simulation time is adjusted in relation to flow rate to obtain roughly equal particle travel lengths. Smaller time increments Δt were necessary at higher flow rates because of certain DEM limiters which otherwise cause the simulation to crash. Write intervals $\Delta t_{w,1}$ and $\Delta t_{w,2}$ were important as these dictated subsequent plot and process resolution. Finer resolution is preferable because data can be extracted at any multiple interval greater than that sampled during simulation. It should be noted that a different output format from that required for post processing, with different write interval is required for visualization. Highest resolution output was produced for case study 3, flow velocity 1 ms^{-1}, and lowest resolution output was produced for case study 1, flow velocity 0.1 ms^{-1}.

It is clear from the data that execution times were dependent on mesh density, time increment and total simulation time. Smaller time steps, longer simulation times and finer meshes contribute to longer simulation runs although some computational time variability may be noted from tabulated data. These are not the only contributors however, bin size which defines that grid density (unrelated to the CFD mesh density) used to compute particle neighbour lists as discussed as part of the theory section, can consume considerable resources. It is also expected that increasing the number of particles in simulation would increase computational time, however the greatest contributor to computation time is expected to be time increment, closely contended by bin size, however this was not investigated nor confirmed.

Tbl. (4.8) *Simulation time statistics*

[a] Sim. Code	[b] Sim. Time	[c] Exec. Time	[d] Clock Time	Δt	$\Delta t_{w,1}$	$\Delta t_{w,2}$
v01f14794	5	9702	14267	0.00002	0.1	0.01
v01f26629	5	9446	14053	0.00002	0.1	0.01
v05f14794	1	5174	7136	0.00001	0.05	0.005
v05f26629	1	4553	6709	0.00001	0.05	0.005
v1f14794	0.5	4000	5039	0.00001	0.02	0.0002
v1f26629	0.5	3552	5171	0.00001	0.02	0.0002

Table notes

[a] *simulation code (v) velocity (01) flow speed 0.1* ms^{-1} *(f) frequency (14794) 14794* Hz

[b] *simulation time in computational seconds*

[c] *CPU execution time (s)*

[d] *wall clock time (s)*

Δt *time increment*

$\Delta t_{w,1}$ *write increment for visualization*

$\Delta t_{w,2}$ *write increment for plotting and data analysis*

Clock times are considerably longer than execution times because of output which is written to file. Large discrepancies between execution and clock times indicate that a considerable portion of simulation time was dedicated to writing output files. Overheads were not as severe in pure CFD flow development simulations, refer to table *4.3*. Computation time was also prolonged by implementing two way coupling where one way coupling would have been adequate. CFD and DEM computation times were not considered in isolation as time was not a restricting factor in this study however in simulations where computation time is a restricting factor analysing computation time characteristics may help identify blockages.

CHAPTER 5

CONCLUSIONS & RECOMMENDATIONS

5.1 Conclusions

The primary objective of this study was to develop a coupled CFD-DEM model incorporating the acoustic force using open source codes OpenFOAM and LIGGGHTS, and to subsequently test the model under various flow rates and frequencies to see if quartz particles ranging between 10 and 60 micron in diameter could be deflected and to what extent this could be achieved. The model was designed and tested on a 20 particle sample of uniform size distribution for three flow rates 0.1, 0.5 and 1.0 ms^{-1} and two driving frequencies 14794 Hz and 26629 Hz. The following conclusions were drawn from simulation results:

5.1.1 CFD

i The flow profile was validated against the Darcy-Weisbach equation through the relation between pressure drop and mean velocity

ii The capped profile characteristic of turbulent pipe and duct flow was recovered from simulation

5.1.2 CFD-DEM

i Particles in the diameter size range 10-60 micron could be deflected at various degrees of efficiency toward respective pressure node planes using an acoustic standing wave field. A critical residence time, for which a particle had to be subject to an acoustic field to attain the required deflection, could be determined for each particle under a given set of acoustic parameters and material properties. It was found that larger particles subject to higher driving frequencies and transported at lower flow rates, were deflected most effectively

ii The study found that the acoustic force was sensitive to driving frequency, particle size and particle node separation distance consistent with the acoustic force equation. Driving frequency was the only adjustable parameter governing wave form nodal spacing, in a given fluid medium. The general trends observed in terms of the effects of frequency, flow rate and particle size are in agreement with both the expected analytic result based on the one dimensional equation form and published experimental findings

iii Larger particles experience stronger acoustic and drag forces. Acoustic forces were found to be on average at least one order of magnitude larger than corresponding drag forces at the lower driving frequency 14794 Hz and about two to three orders of magnitude greater than drag forces at the higher driving frequency 26629 Hz, based on initial values which quickly became damped. Particle material density in the limited range investigated seemed to have had a relatively minor contribution to the acoustic force magnitude, however did determine the sign of the acoustic contrast factor which resulted in the denser solid particles being forced to nodal planes, consistent with literature

iv The required transducer length was dependent on the strength of the acoustic force in relation to the particulate velocity, typically that of the carrier fluid at the particle coordinate. Higher flow velocities called for longer acoustic lengths, to achieve the required deflections to within the 1.5 mm tolerance distance. Critical transducer lengths required to deflect 4 of 6 sampled particles varied between 5 and 45 cm when driven at 14794 Hz and between 2 and 12 cm when driven at 26629 Hz depending on the flow profile. All but one particle could be deflected sufficiently at the higher driving frequency 26629 Hz over the given travel length whereas the lower frequency 14794 Hz proved ineffective in deflecting smaller particles radially sized 5-10 micron. In all cases smaller particles responded poorly to the acoustic force

v Under the investigated set of parameters and simplifying assumptions, the system did not produce any industrially implementable results, in terms of not only deflection separation distance but also in terms of neglected physics

vi Lower driving frequencies were found better suited to producing practically spaced pressure node fields, however would have to be supplemented by higher driving amplitudes to produce practical deflecting strength. The achievable amplitudes would be determined by transducer design. High frequency systems are more commonly used in promoting agglomeration as a pre-filtering method

5.1.3 Computational Aspects

CFD

i The CFD log of the wall profile was recovered, and y parameters were found comparable in the logarithmic layer, consistent with literature

ii The implemented RANS $k-\epsilon$ model could not simulate vorticular and secondary flow phenomena

iii Execution time was found to be largely dependent on mesh density and less so on simulation time. Time overheads were found to be considerably less in pure CFD runs than in coupled runs

CFD-DEM

- **i** Particles experience large initial z drag and x acoustic forces in the absence of damping. Damping forces are invoked following the first time step in response to particle motion, resulting in order of magnitude force reductions in either direction. Subsequently particles converge toward their equilibrium nodal plane coordinates tending to machine precision in the absence of disruptive forces. The response was attributed to viscous drag acting in opposition to inertial velocity which prevented persistent non-physical particle oscillation about nodal planes
- **ii** Acoustic and drag forces were found to be partially decoupled in the sense that one did not affect the other directly, however acoustic forces did cause lateral deflection of particles and since the flow field was spatially variant in the cross sectional plane, different z force and velocity components were registered by individual particles under different driving frequencies
- **iii** Coupled simulation clock time was found to be dependent on mesh density, time increment, simulation time, DEM particle bin size and output write resolution. A considerable portion of total clock time was attributable to writing output files

5.2 Recommendations

The model designed in this study was flexible in terms of parameter variation which allows a variety of systems to be investigated without excessive overhaul. It is easily extendible to include additional forces either between particles or between fluid and particles, or those arising from external drivers. The model is also expected to be directly applicable to small scale systems without modification, other than the necessary geometric changes, as the presently unaccounted for physics of acoustic non-linearity and turbulent eddying are absent in such systems.

There are a number of additions which could be incorporated into the model to account for physics which had otherwise been omitted in the current study under the adopted set of assumptions. The effect of turbulent eddies and secondary flows on the particulate system could be modelled through stochastic methods introduced through CFDEM force models. Alternatively DNS or finely meshed LES simulations could produce the resolved eddy scales which would interact with small particles. Secondary flows are resolved by RSM's, the implementation of which could serve as the first transition from the current model. Turbulent eddies and secondary flows are expected to disperse the particulate phase which would result in reduced acoustic deflection efficiency.

The assumption of an ideal standing wave field in a large cavity such as the one used in the study, is inaccurate. Acoustic energy is absorbed in any medium through which it travels and is dissipated as heat energy. Transducers which have been designed for this purpose would produce far less wave attenuation, however it cannot be completely eliminated. In addition to attenuation, reflection should not be assumed perfect although with the use of an appropriate reflecting wall material and careful overall design, near perfect reflection can be achieved. Attenuation and real reflection result in acoustic non-linearities which may be accounted for analytically in simple geometric cavities and numerically in more irregular cavities. Irregularly shaped cavities would further produce interference pressure wave forms which would require a more involved form of the acoustic force equation than the one

used in this study. A pressure field which has been obtained either numerically or analytically may be imposed over a flow field from which forces acting on particles may be computed based on local pressure gradients. As this force would be more pressure field based a user defined force model to account for it may be included in the CFDEM code, which would allow access to all fluid fields as necessary including pressure. Alternatively the pressure field could be computed separately and using point data to introduce the force directly on the DEM side.

The current study could be improved by inserting more particles and reducing time increment to resolve collisions, again a time consuming exercise which may or may not have to be run on a cluster depending on the number of particles inserted. The case could be run as a one way coupling process which may improve computation time considerably provided the number of particles does not invalidate dilute system criteria. The particle seed could be varied to produce different configurations which could be investigated from a statistical point of view. Additional inter-particle short and long range forces as well as cohesion forces may become relevant under a smaller time step, the analytic forms of which have to be determined based on the nature of the interacting particles. Studies of this type could prove valuable in investigating agglomeration phenomena. Increasing the number of particles could push the system into a dense two or four way coupling regime which may require revision in terms of modelling assumptions and case setup. In addition to time step, write interval should also be reduced. Write interval is important for plotting data at adequate resolution especially in cases where sharp transients manifest and contacts are of interest.

The acoustic effect on gas bubbles or particles with a density ratio in favour of the fluid phase could be investigated. These low density particles would in most cases be deflected toward anti-nodal planes. Typically gas bubbles are larger than solid particles in industrial systems, and require additional consideration with respect to contact properties, surface tension and deformation. Small rigid bubbles may be modelled in DEM as spherical particles with a certain set of material and interaction properties which would attempt to mimic those of a physical bubble. Larger deformable bubbles may be better modelled using multiphase CFD, in which case only a single bubble can be modelled, as the gas phase is considered a continuum. The acoustic force on the bubble would then have to be treated on the CFD side. Another scenario of interest may be that of bubble particle interaction within an acoustic field, in which case relevant bubble particle contact forces would have to be defined. Depending on how the bubble is modelled interactions have to be specified either in the DEM or CFDEM frameworks. Particle interaction is resolved by small time increments, warranting the need for time consuming parallel cluster runs.

BIBLIOGRAPHY

Articles

[1] Y. Abe, M. Kawaji, and T. Watanabe. Study on the bubble motion control by ultrasonic wave. *Experimental Thermal and Fluid Science*, 26(6-7):817–826, 8 2002.

[2] M. Banks, N. T. Bridges, and M. Benzit. Measurements of the Coefficient of Restitution of Quartz Sand on Basalt: Implications for Abrasion Rates on Earth and Mars. In S. Mackwell and E. Stansbery, editors, *36th Annual Lunar and Planetary Science Conference, March 14-18, 2005, in League City, Texas, abstract no.2116*, volume 36, page 2116, Mar. 2005.

[3] M. Barmatz. Acoustic radiation potential on a sphere in plane, cylindrical, and spherical standing wave fields. *The Journal of the Acoustical Society of America*, 77:928, 1985.

[4] E. R.-F. de Sarabia. Application of high-power ultrasound to enhance fluid/solid particle separation processes. *Ultrasonics*, 38(1):642, 2000.

[5] A. Doehle, H. Landes, J. Hoffelner, R. Lerch, and M. Kristen. Ultrasonic standing waves in industrial applications. In *Ultrasonics, 2003 IEEE Symposium on*, volume 2, pages 2045 – 2048 Vol.2, oct. 2003.

[6] A. A. Doinikov. Effects of the second harmonic on the secondary bjerknes force. *Phys. Rev. E*, 59:3016–3021, Mar 1999.

[7] H. D. H. Drane. Elastic constants of fused quartz. change of young's modulus with temperature. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, 122(789):pp. 274–282, 1929.

[8] J. Gallego-Juarez. New technologies in high-power ultrasonic industrial applications. In *Ultrasonics Symposium, 1994. Proceedings., 1994 IEEE*, volume 3, pages 1343 –1352 vol.3, 31 1994-nov. 3 1994.

[9] C. Geuzaine and J.-F. Remacle. Gmsh: A 3-d finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331, 2009.

[10] S. Gupta, D. L. Feke, and I. Manas-Zloczower. Fractionation of mixed particulate solids according to compressibility using ultrasonic standing wave fields. *Chemical Engineering Science*, 50(20):3275–3284, 10 1995.

[11] J. Hawkes, J. Cefai, D. Barrow, W. Coakley, and L.Briarty. Ultrasonic manipulation of particles in microgravity. *Journal of physics.D, Applied physics*, 31(14):1673, 1998.

[12] J. J. Hawkes and W. T. Coakley. A continuous flow ultrasonic cell-filtering method. *Enzyme and microbial technology*, 19(1):57–62, 7 1996.

[13] J. J. Hawkes and W. T. Coakley. Force field particle filter, combining ultrasound standing waves and laminar flow. *Sensors and Actuators B: Chemical*, 75(3):213–222, 5/15 2001.

[14] M. Hill, J. J. Hawkes, and Y. Shen. Modelling of layered resonators for ultrasonic separation. *Ultrasonics*, 40(1-8):385, 2002.

[15] D. A. Johnson and D. L. Feke. Methodology for fractionating suspended particles using ultrasonic standing wave and divided flow fields. *Separations Technology*, 5(4):251–258, 11 1995.

[16] J. A. G. Juarez. A macrosonic system for industrial processing. *Ultrasonics*, 38(1):331, 2000.

[17] M. Kaltenbacher, H. Landes, J. Hoffelner, and R. Simkovics. Use of modern simulation for industrial applications of high power ultrasonics. In *Ultrasonics Symposium, 2002. Proceedings. 2002 IEEE*, volume 1, pages 673 – 678 vol.1, oct. 2002.

[18] S. S. Kim, H. J. Hwang, M. W. Shin, and H. Jang. Friction and vibration of automotive brake pads containing different abrasive particles. *Wear*, 271(7 - 8):1194 – 1202, 2011.

[19] L. V. King. On the acoustic radiation pressure on spheres. *Proceedings of the Royal Society of London. Series A, Mathematical and Physical Sciences*, 147(861):pp. 212–240, 1934.

[20] C. Kloss, C. Goniva, G. Aichinger, and S. Pirker. Comprehensive dem-dpm-cfd simulations-model synthesis, experimental validation and scalability. In *Seventh International Conference on CFD in the Minerals and Process Industries*, page 1. CSIRO, 9-11 Dec 2009.

[21] C. Kloss, C. Goniva, A. Hager, S. Amberger, and S. Pirker. Models, algorithms and validation for opensource dem and cfd-dem. *Progress in Computational Fluid Dynamics, an International Journal*, 12(2):140–152, 01 2012.

[22] Y. Li, Y. Xu, and C. Thornton. A comparison of discrete element simulations and experiments for sandpiles composed of spherical particles. *Powder Technology*, 160(3):219 – 228, 2005.

[23] E. Loth and A. J. Dorgan. An equation of motion for particles of finite reynolds number and size. *Environmental fluid mechanics*, 9(2):187–206, 2009.

[24] M. Nallasamy. A critical evaluation of various turbulence models as applied to internal fluid flows. *NASA STI/Recon Technical Report N*, 852:25757, 1985.

[25] OpenFOAM. *"An Open Source CFD-DEM Perspective"*, Göteborg, June 22.-24. 2010.

[26] S. V. Pepper. Shear strength of metal - SiO2 contacts. In *American Physical Society, U.S. Navy, and ARPA, International Topical Conference on the Physics of SiO2 and Its Interfaces, Yorktown Heights, N.Y ; United States*, volume 5, Mar. 1978.

[27] S. Plimpton. Fast parallel algorithms for short-range molecular dynamics. *Journal of Computational Physics*, 117(1):1 – 19, 1995.

[28] S. Plimpton, R. Pollock, and M. Stevens. Particle-mesh ewald and rrespa for parallel molecular dynamics simulations. In *PPSC, In Proceedings of the Eighth SIAM Conference on Parallel Processing for Scientific Computing*, 1997.

[29] K. C. Radil and A. B. Palazzolo. Influence of temperature and impact velocity on the coefficient of restitution. *Unknown*, 95:10813, July 1994.

[30] A. Sarvazyan and L. Ostrovsky. Stirring and mixing of liquids using acoustic radiation force. *The Journal of the Acoustical Society of America*, 125:3548, 2009.

[31] M. E. Smith. Wave phenomena in an acoustic resonant chamber. *American journal of physics*, 42:131, 1974.

[32] M. Sommerfeld and N. Huber. Experimental analysis and modelling of particle-wall collisions. *International Journal of Multiphase Flow*, 25(6 - 7):1457 – 1489, 1999.

[33] J. F. Spengler, W. T. Coakley, and K. T. Christensen. Microstreaming effects on particle concentration in an ultrasonic standing wave. *AICHE Journal*, 49(11):2773–2782, 2003.

[34] J. F. Spengler, M. Jekel, K. T. Christensen, R. J. Adrian, J. J. Hawkes, and W. T. Coakley. Observation of yeast cell movement and aggregation in a small-scale mhz-ultrasonic standing wave field. *Bioseparation*, 9:329–341, 2001.

[35] C. Speziale, R. So, and B. Younis. On the prediction of turbulent secondary flows. Technical report, DTIC Document, 1992.

[36] G. D. Toro. Friction falls towards zero in quartz rock as slip velocity approaches seismic rates. *Nature*, 427(6973):436, 2004.

[37] R. J. Townsend, M. Hill, N. R. Harris, and N. M. White. Modelling of particle paths passing through an ultrasonic standing wave. *Ultrasonics*, 42(1-9):319–324, 4 2004.

[38] C. Vanhille and C. Campos-Pozuelo. Nonlinear ultrasonic waves in bubbly liquids with nonhomogeneous bubble distribution: Numerical experiments. *Ultrasonics Sonochemistry*, 16(5):669 – 685, 2009.

[39] C. Vanhille, C. Conde, and C. Campos-Pozuelo. Finite-difference and finite-volume methods for

nonlinear standing ultrasonic waves in fluid media. *Ultrasonics*, 42(19):315 – 318, 2004.

[40] H. G. Weller, G. Tabor, H. Jasak, and C. Fureby. A tensorial approach to computational continuum mechanics using object-oriented techniques. *Computers in Physics*, 12:620, 1998.

[41] G. Whitworth and W. T. Coakley. Particle column formation in a stationary ultrasonic field. *Journal of the Acoustic Society of America*, 91(1), 1992.

Books

[42] O. Corporation. *OpenFOAM The Open Source CFD Toolbox Programmers Guide*. OpenFOAM, 59 Temple place, Suite 330, Boston, MA 02111-1307 USA, 1.7.1 edition, August 2010.

[43] O. Corporation. *OpenFOAM The Open Source CFD Toolbox User Guide*. OpenFOAM, 59 Temple place, Suite 330, Boston, MA 02111-1307 USA, 2.0.0 edition, June 2011.

[44] L. Fan and C. Zhu. *Principles of Gas-Solid Flows*. Cambridge Series in Chemical Engineering. Cambridge University Press, 2005.

[45] Fluent. *Fluent 6.2 User Guide*. Fluent Inc, Centerra Resource Park, 10 Cavendish Court, Lebanon, NH 03766, USA, 2005.

[46] C. Geuzaine and J.-F. Remacle. *Gmsh Reference Manual 2.5*. Dept. of Electrical Engineering and Computer Science, University of Liège, Montefiore Institute, Sart-Tilman, Bldg. B28, B-4000 Liège, Belgium, October 2010.

[47] A. Hancock. Observation of forces on microparticles in acoustic standing waves. Master's thesis, University of California, 2001.

[48] S. Hardt and F. Scheinfeld. *Microfluidic Technologies for Miniaturized Analysis Systems*. Springer Science & Business Media, LLC, New York, 2007.

[49] R. C. Hibbeler. *Engineering Mechanics - Dynamics (11th Edition)*. Prentice Hall, 11 edition, May 2006.

[50] L. E. Kinsler and A. R. Frey. *Fundementals of Acoustics*. John Wiley & Sons, Ltd, New York, London, 2nd edition, 1962.

[51] J. L. Malers. *Fundamentals of Bubble Transport in an Ultrasonically Assisted Separation Process*. PhD thesis, Case Western Reserve University, 2008.

[52] T. Pöschel and T. Schwager. *Computational granular dynamics : models and algorithms*. Springer-Verlag, 2005.

[53] A. Salman, M. Ghadiri, and M. Hounslow. *Particle Breakage*. Handbook of Powder Technology. Elsevier, 2007.

[54] S.Plimpton. *LAMMPS Users Manual*. Sandia National Laboratories, 1515 Eubank Boulevard, Southeast Albuquerque, NM 87185 USA, 2003.

[55] F. White. *Fluid mechanics*. McGraw-Hill, 1979.

[56] F. White. *Viscous fluid flow*. McGraw-Hill series in mechanical engineering. McGraw-Hill, 1991.

[57] G. Wierink. *A computational framework for coupled modelling of three-phase systems with soluble surfactants*. PhD thesis, Alto University, 2012.

Web Sources

[58] http://www.sengpielaudio.com/StandingWaves.html.

[59] http://www.acs.psu.edu/drussell/Demos/superposition/superposition.html#standing.

[60] http://openfoamwiki.net/index.php/Tutorials_for_particle_based_methods.

[61] http://nf.nci.org.au/facilities/software/LIGGGHTS/doc.

[62] http://en.wikipedia.org/wiki/Silicon_dioxide.

[63] http://en.wikipedia.org/wiki/Silicon.

[64] http://www.azom.com/properties.aspx?ArticleID=1114.

[65] http://www.memsnet.org/material/siliconsibulk/.

[66] http://www.matweb.com/.

[67] http://www.stanford.edu/Some_Useful_Numbers.

[68] http://hypertextbook.com/facts/2005/steel.shtml.

[69] http://en.wikipedia.org/wiki/Centimetre_gram_second_system_of_units.

INDEX

LIST OF ABBREVIATIONS

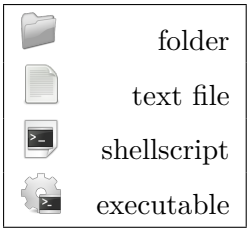
ATS	Adaptive Time Stepping
BASF	Baden Aniline and Soda Factory
BEM	Boundary Element Method
CAD	Computer Aided Design
CDLPFM	Christian Doppler Laboratory on Particulate Flow Modelling
CERECAM	Centre for Research in Computational and Applied Mechanics
CFD	Computational Fluid Dynamics
CFDEM	Computational Fluid Dynamics coupled to Discrete Element Method
CGS	Centimetre Gram Second
COF	Coefficient Of Friction
COR	Coefficient of Restitution
CPU	Central Processing Unit
dB	decibel
DEM	Discrete Element Method
DIC	Diagonal Incomplete Cholesky
DILU	Diagonal Incomplete Lower Upper
DLVO	Derjaguin, Landau, Verwey and Overbeek
DNS	Direct Numerical Simulation
DOF	Degrees Of Freedom
FEM	Finite Element Method
GAMG	Generalized Geometric-Algebraic Multi-Grid
Gmsh	G Mesher
GUI	Graphical User Interface
JKU	Johannes Kepler University
LAMMPS	Large-scale Atomic Molecular Massively Parallel Simulator
LES	Large Eddy Simulation
LHS	Left Hand Side
LIGGGHTS	LAMMPS Improved for General Granular and Granular Heat Transfer Simulations
LRR	Launder Rodi Reece
LTS	Long Term Support
MatLab	Matrix Laboratory
MPI	Message Passing Interface
NSE	Navier-Stokes Equations
OpenFOAM	Open source Field Operation and Manipulation
ParaView	Parallel Visualization Application
PBiCG	Preconditioned Bi-Conjugate Gradient
PCG	Preconditioned Conjugate Gradient
PISO	Pressure-Implicit Split-Operator

PZT	Lead Zirconate Titanate
RANS	Reynolds Averaged Navier Stokes
RHS	Right Hand Side
RLC	Resistor Inductor Capacitor
RSM	Reynolds Stress Model
SI	International Metric
STL	Standard Tessellation Language
TVD	Total Variation Diminishing
UCT	University of Cape Town
USA	United States of America
VOF	Volume Of Fluid
VTK	Visualization Tool kit

APPENDIX A

FIGURES

Case Directory Structure



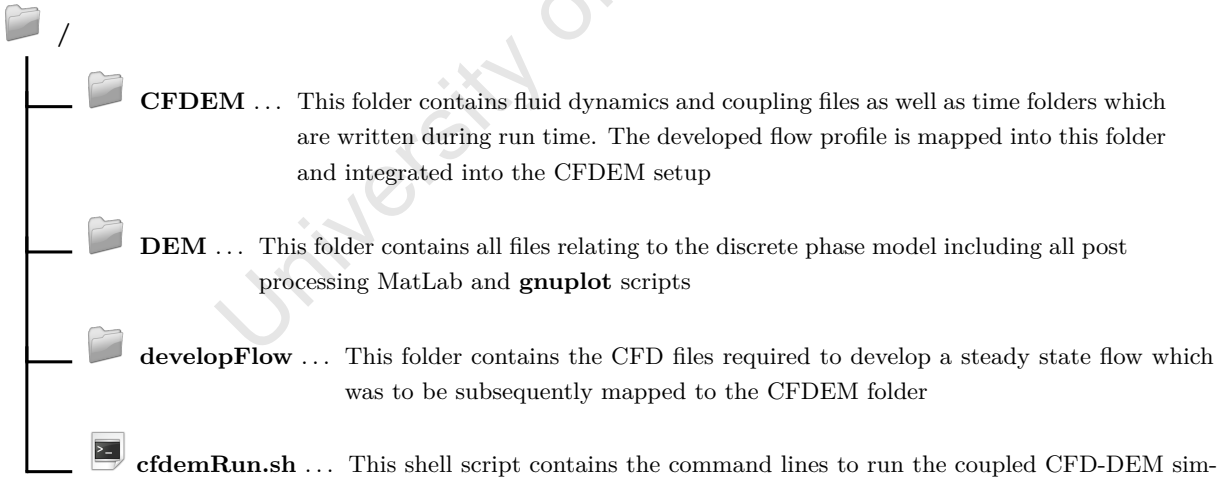


Fig. (A.1) *Coupled case directory*

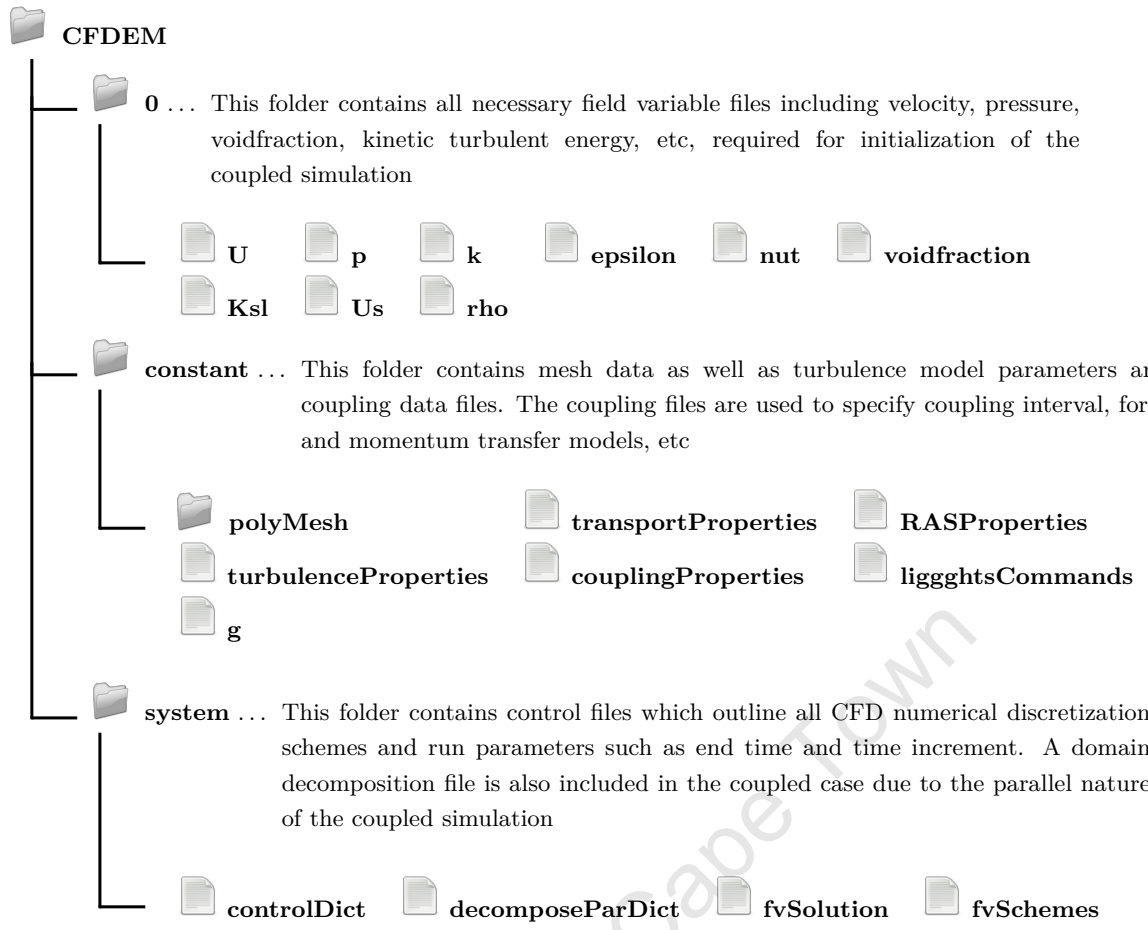


Fig. (A.2) *Coupled case CFDEM directory*

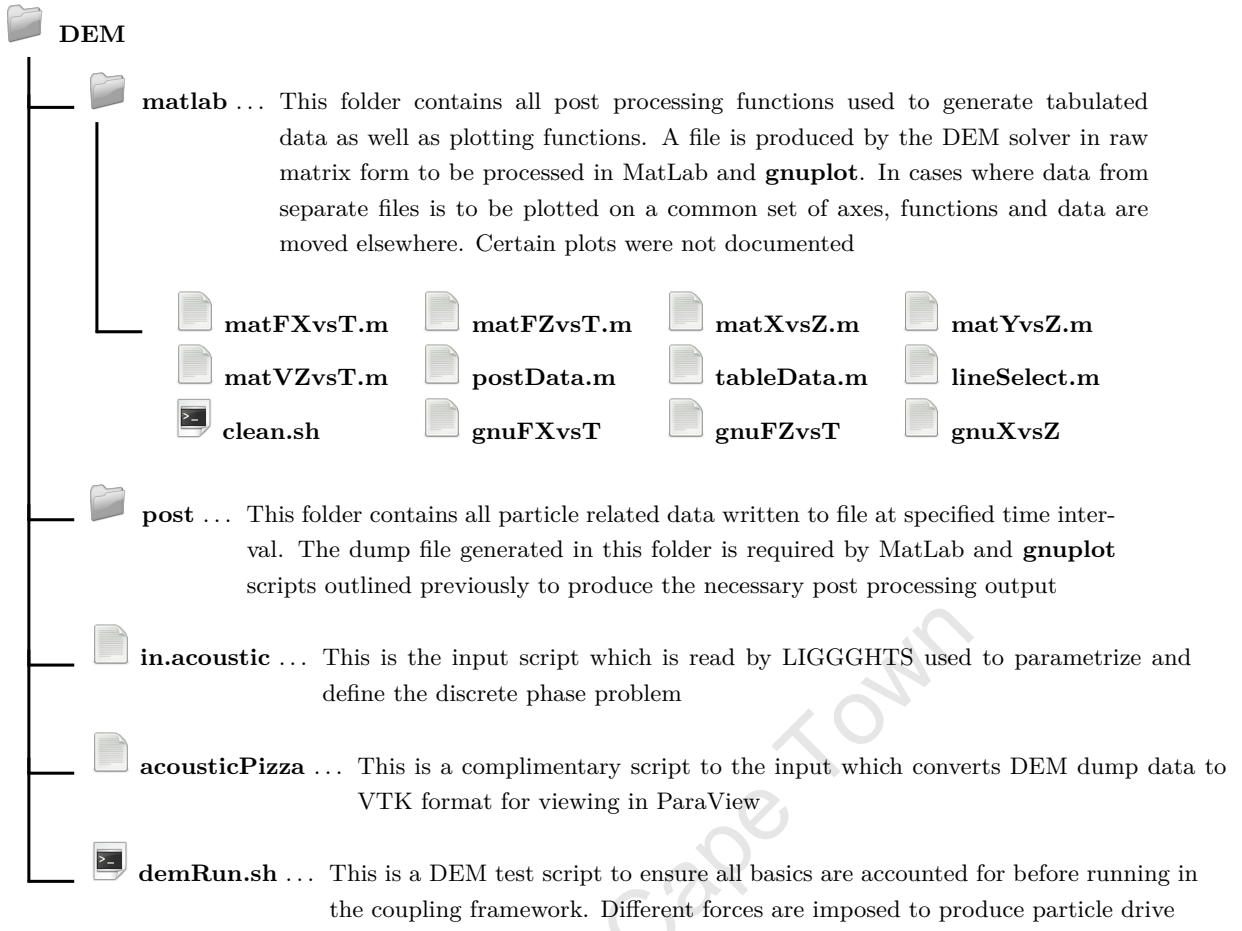


Fig. (A.3) *Coupled case DEM directory*

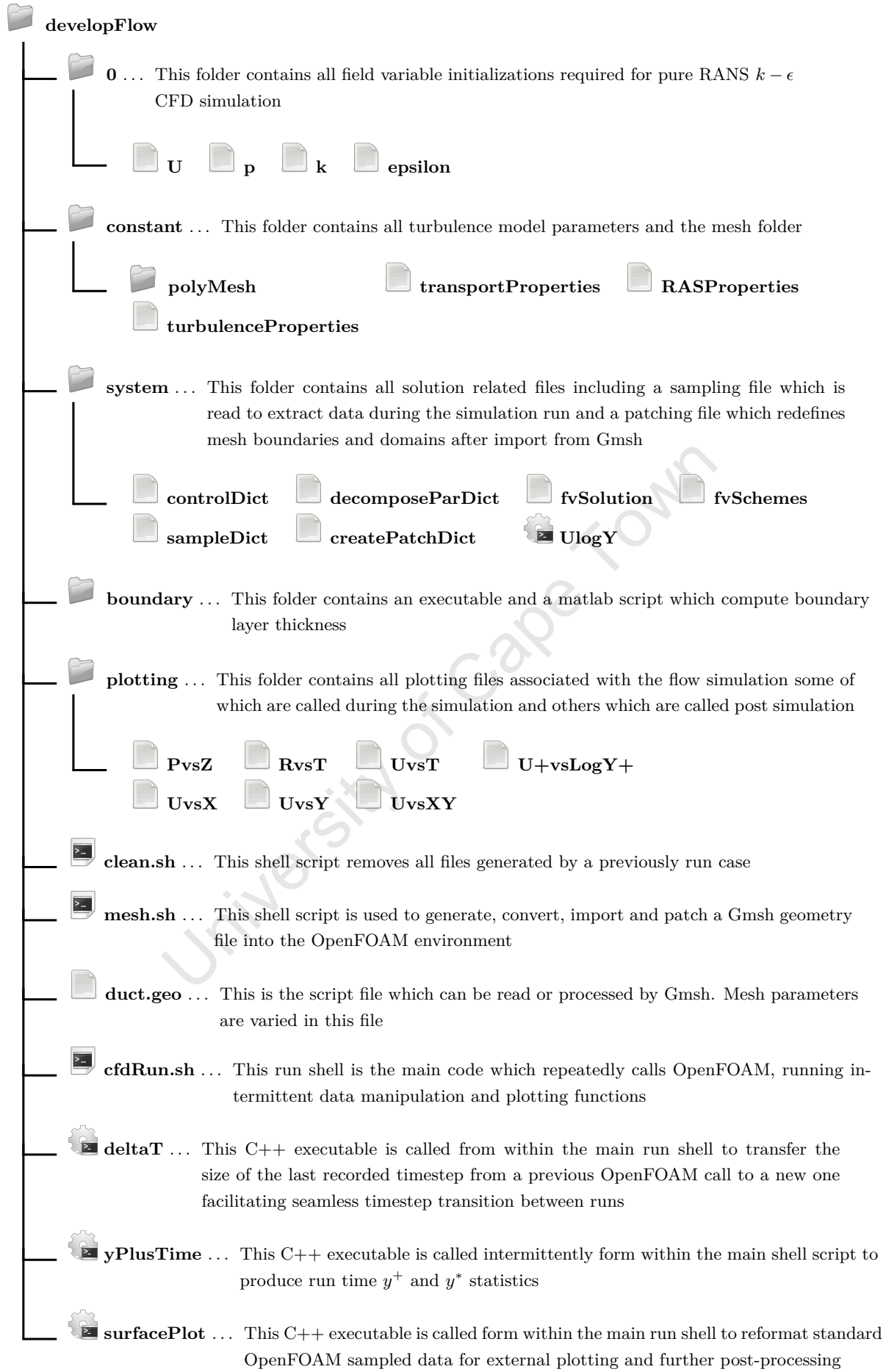
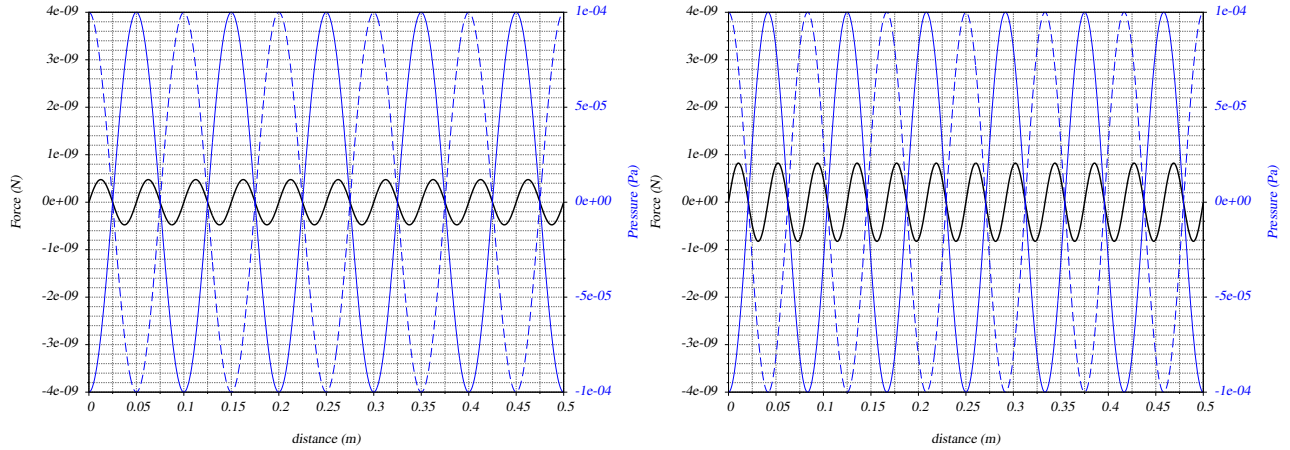


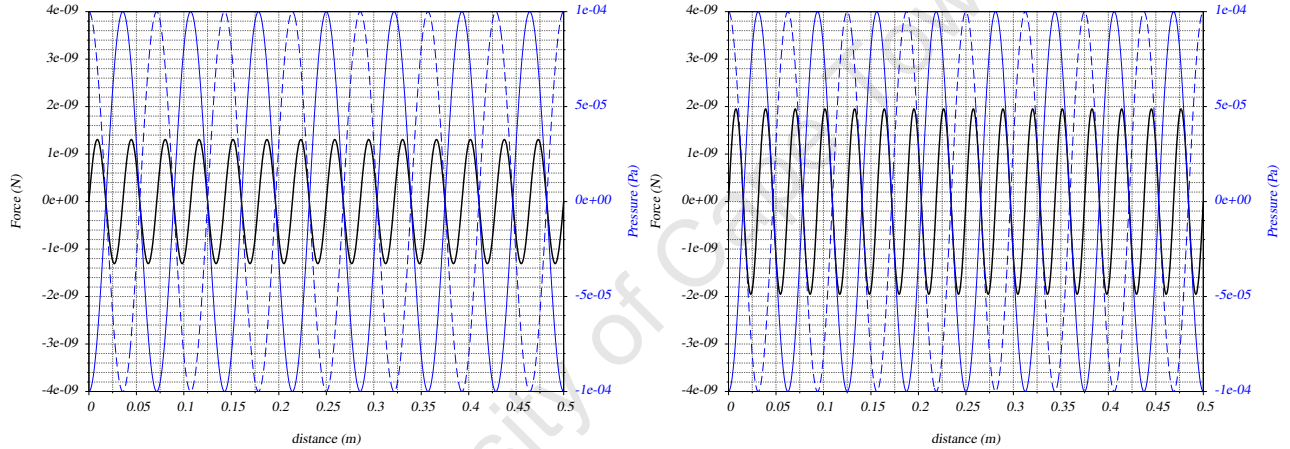
Fig. (A.4) *Coupled case CFD directory*

Analytic Fields



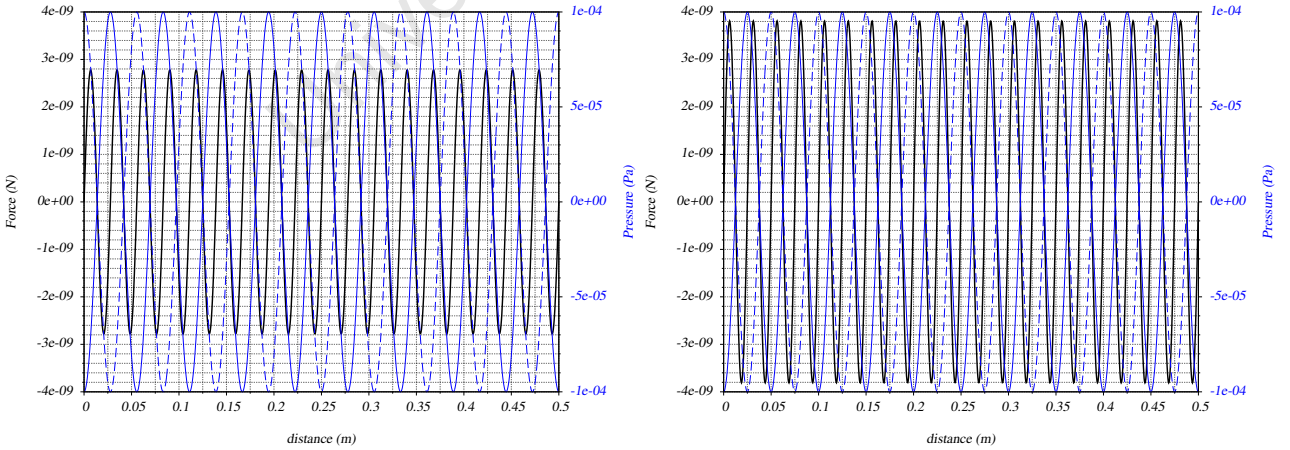
(a) f=14794, n=10

(b) f=17753, n=12



(c) f=20712, n=14

(d) f=23670, n=16



(e) f=26629, n=18

(f) f=29588, n=20

Fig. (A.5) *Analytic force and pressure fields for the ideal case* $A = 50\mu m$, $r = 17.5\mu m$, $\rho_p = 2500\, kgm^{-3}$

Case Study 2 $U_{max} = 0.5ms^{-1}$

Flow Development

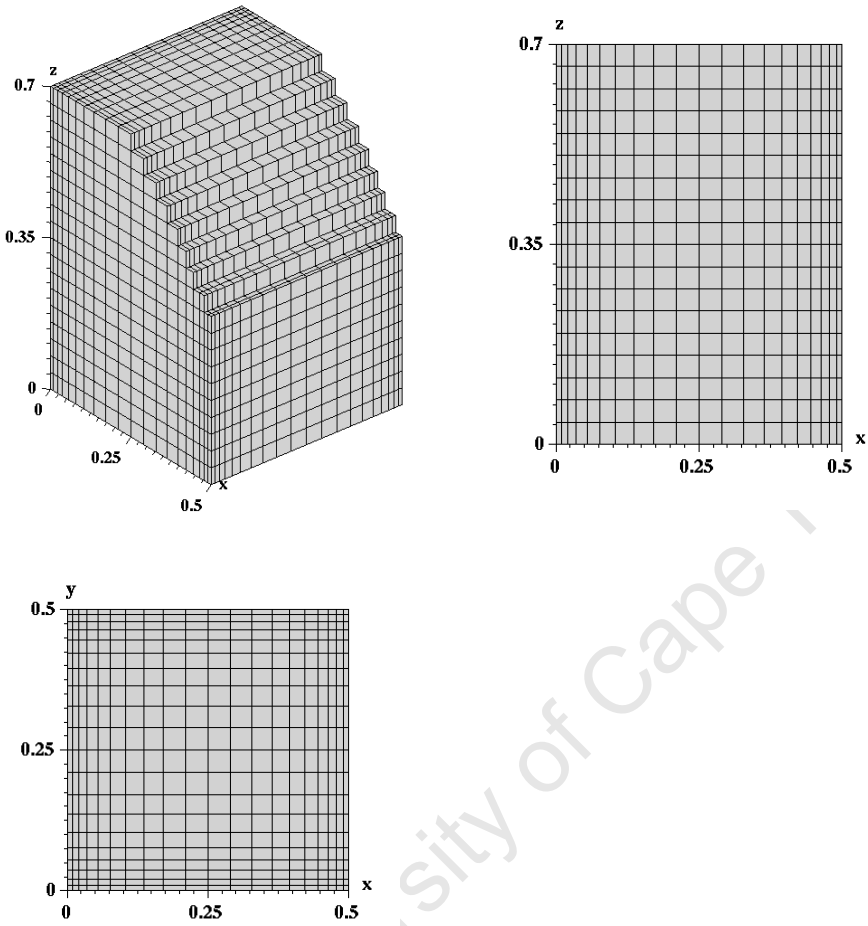
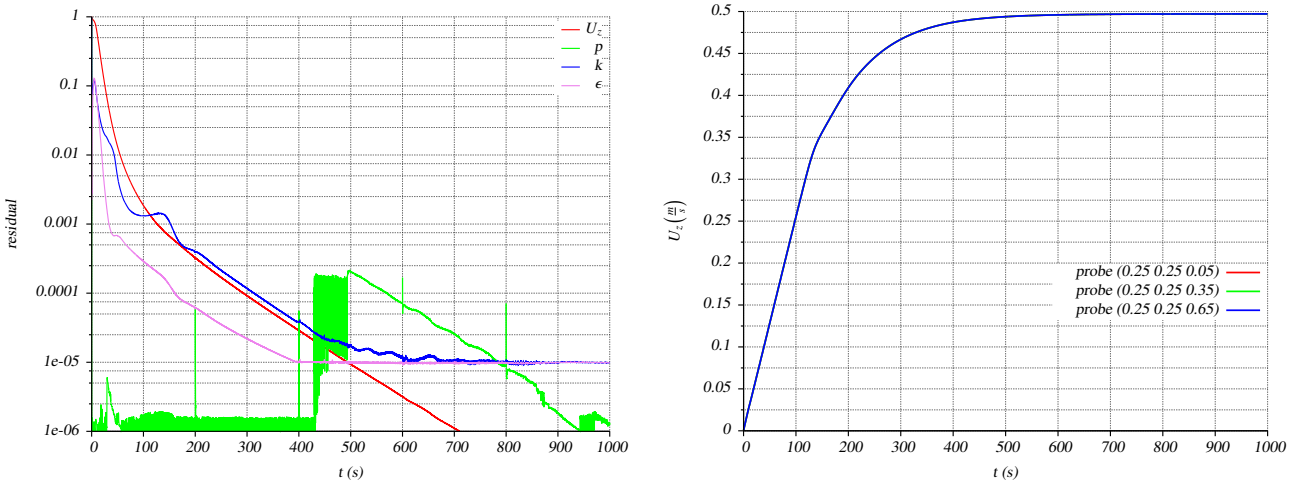
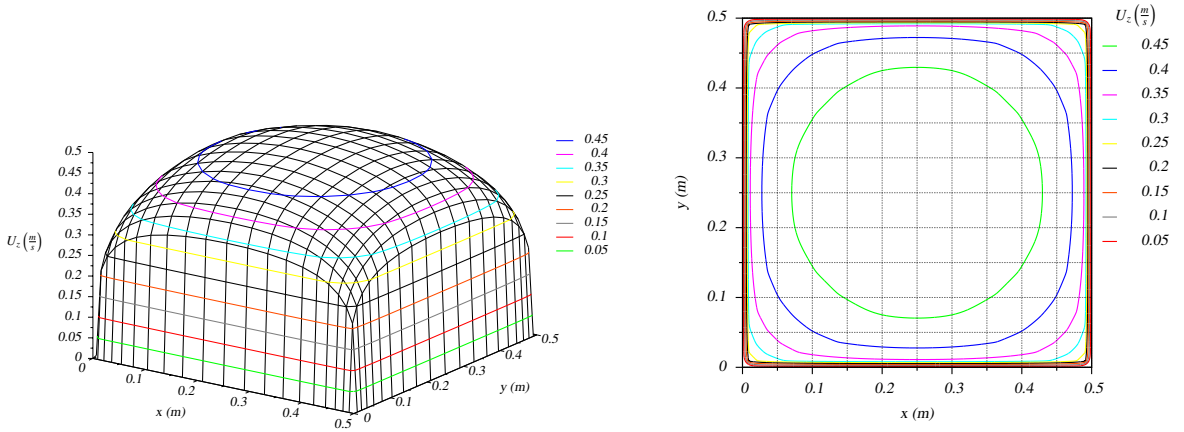


Fig. (A.6) *Mesh elements: 7200, (n_x, n_y, n_z) = (20, 20, 18), wall graded r=0.2*



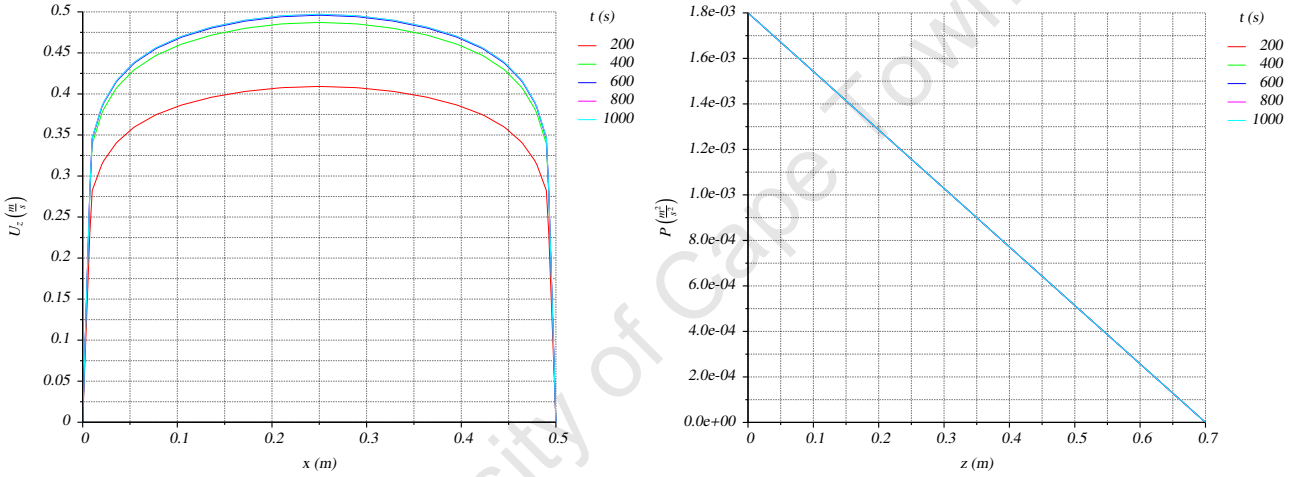
(a) *Residuals* ***(b)*** *U_z vs t*

Fig. (A.7) *Solution convergence*



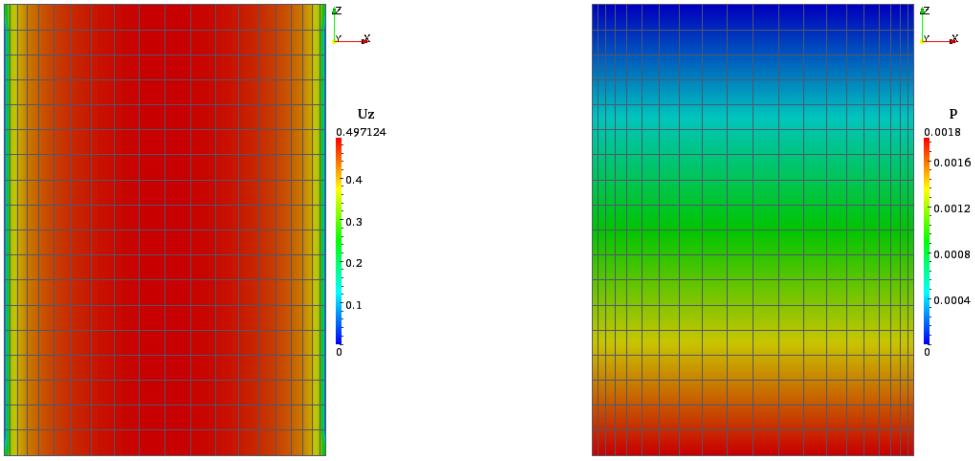
(a) U_z *vs xy, mesh grid* *(b)* U_z *vs xy, contours*

Fig. (A.8) *Cross sectional flow*



(a) U_z *vs x* *(b)* P *vs z*

Fig. (A.9) *Velocity and pressure profile evolution*



(a) Velocity field *(b) Pressure field*

Fig. (A.10) *Developed velocity and pressure fields*

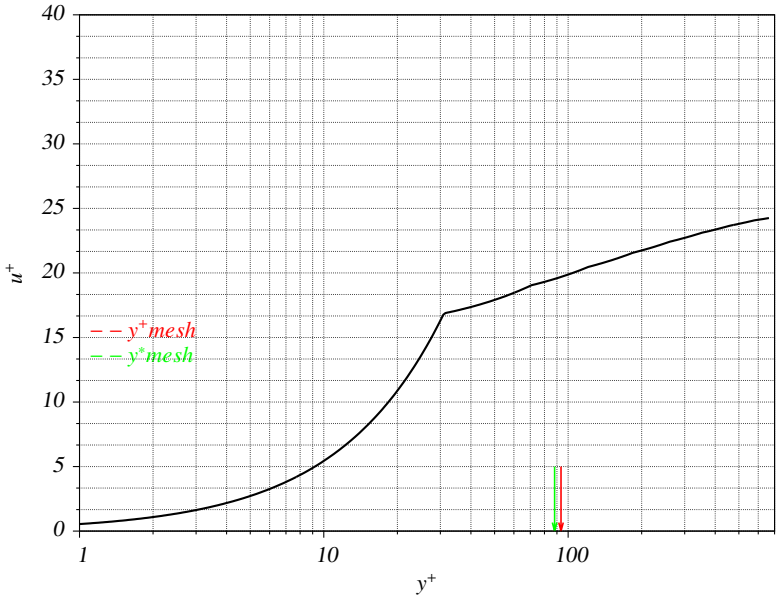
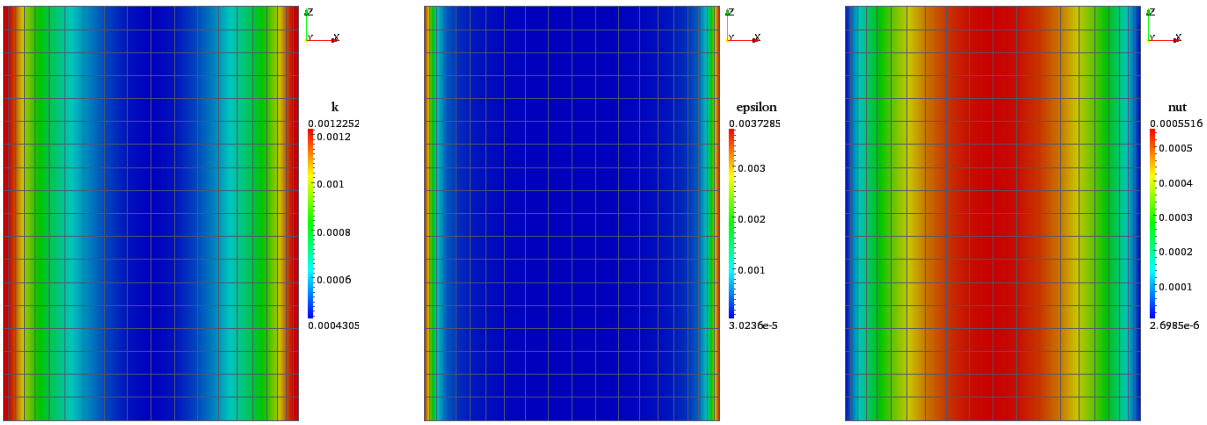


Fig. (A.11) *Log of the wall profile*



(a) *k Field* ***(b)*** *ϵ Field* ***(c)*** *μ_t Field*

Fig. (A.12) *Kinetic energy, dissipation rate and turbulent viscosity fields, xz slice plane*

Case Study 3 $U_{max} = 1.0ms^{-1}$

Flow Development

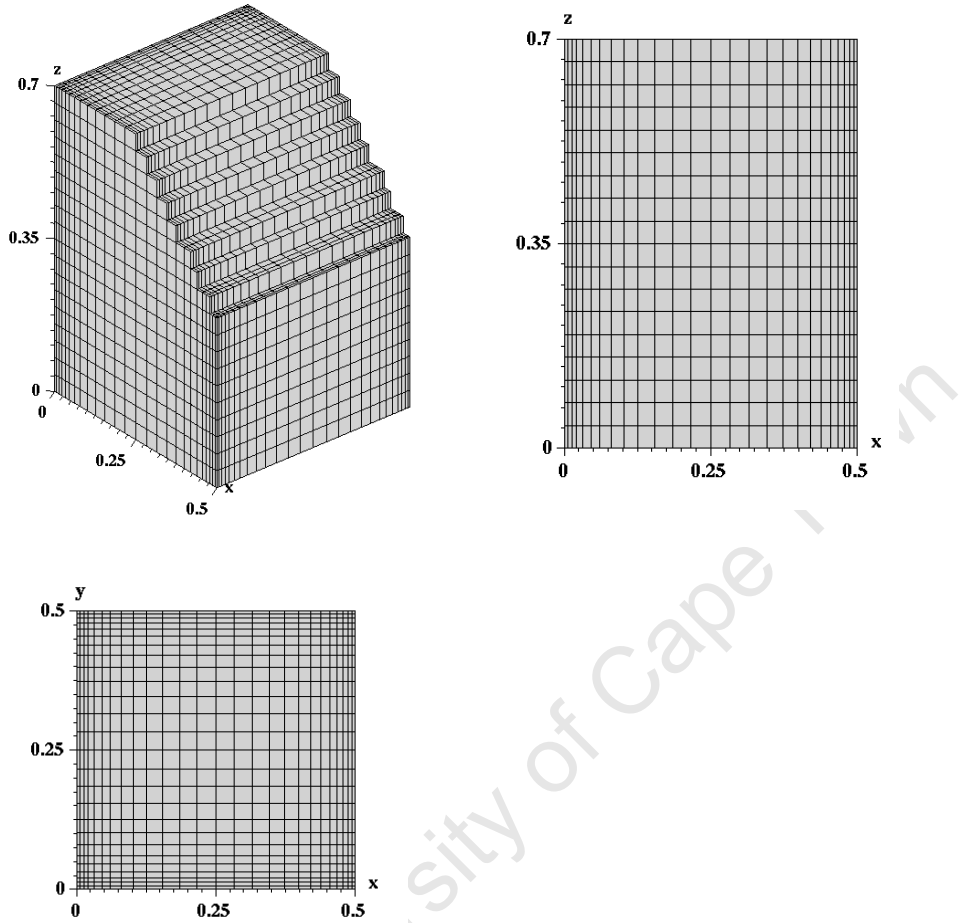
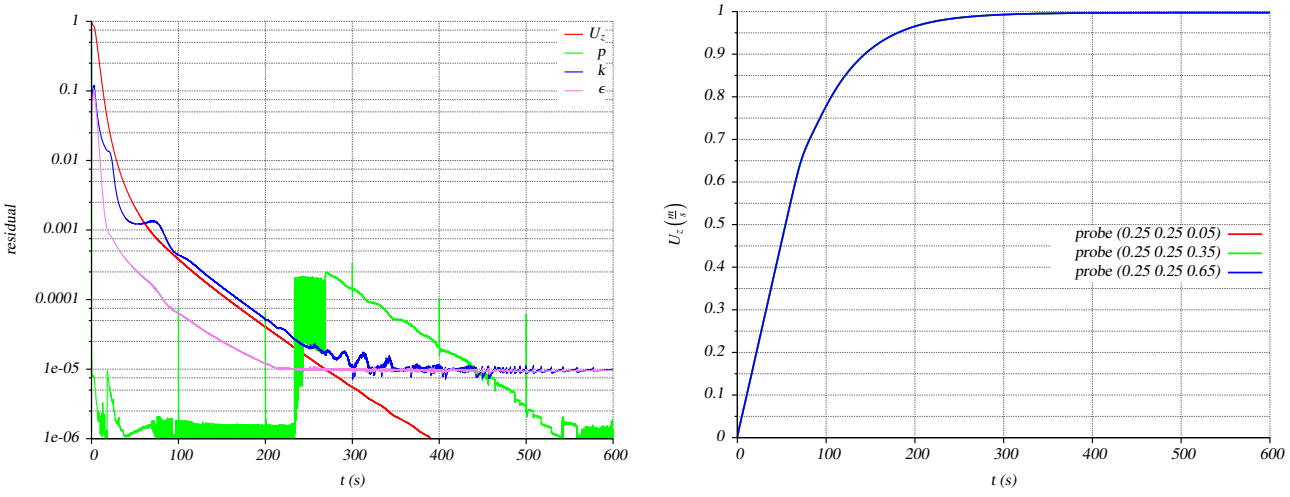
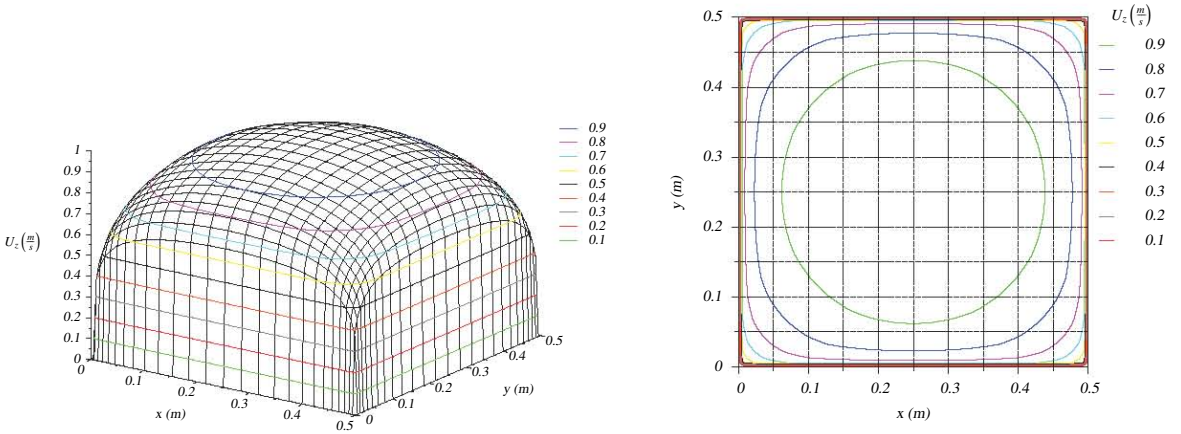


Fig. (A.13) *Mesh elements: 12168, (n_x, n_y, n_z) = (26, 26, 18), wall graded r=0.15*



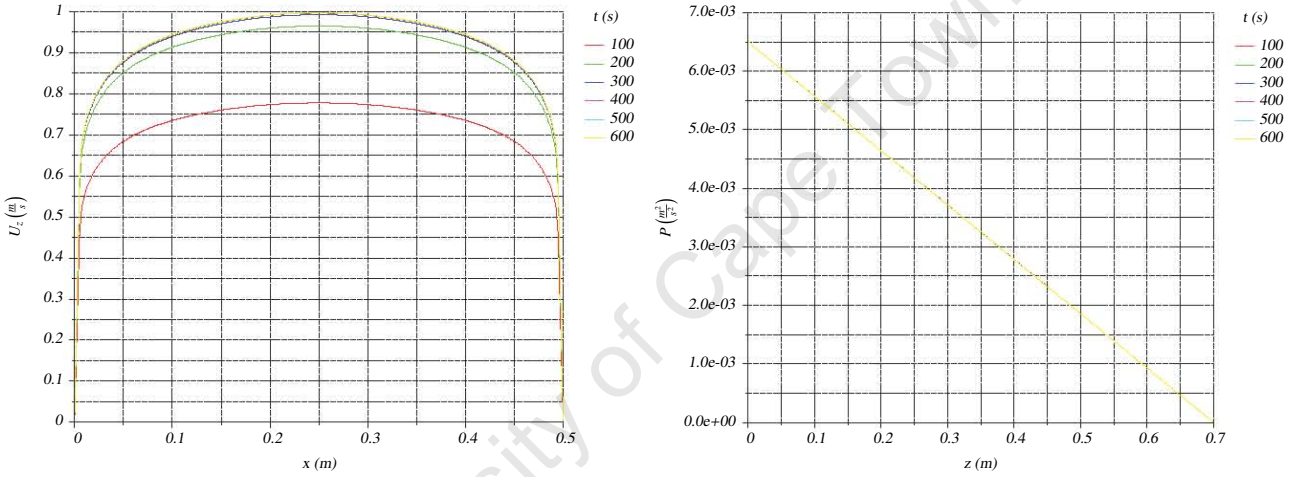
(a) *Residuals* ***(b)*** *U_z vs t*

Fig. (A.14) *Solution convergence*



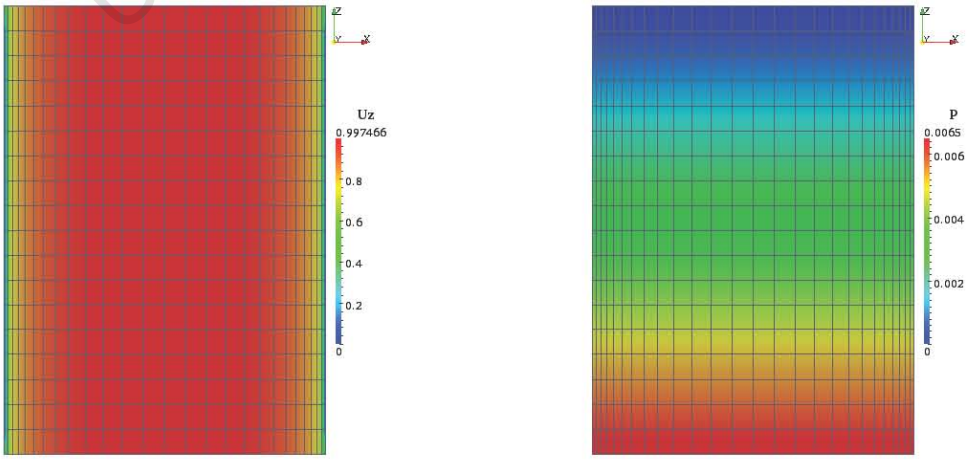
(a) U_z *vs xy, mesh grid* *(b)* U_z *vs xy, contours*

Fig. (A.15) *Cross sectional flow*



(a) U_z *vs x* *(b)* P *vs z*

Fig. (A.16) *Velocity and pressure profile evolution*



(a) Velocity field *(b) Pressure field*

Fig. (A.17) *Developed velocity and pressure fields*

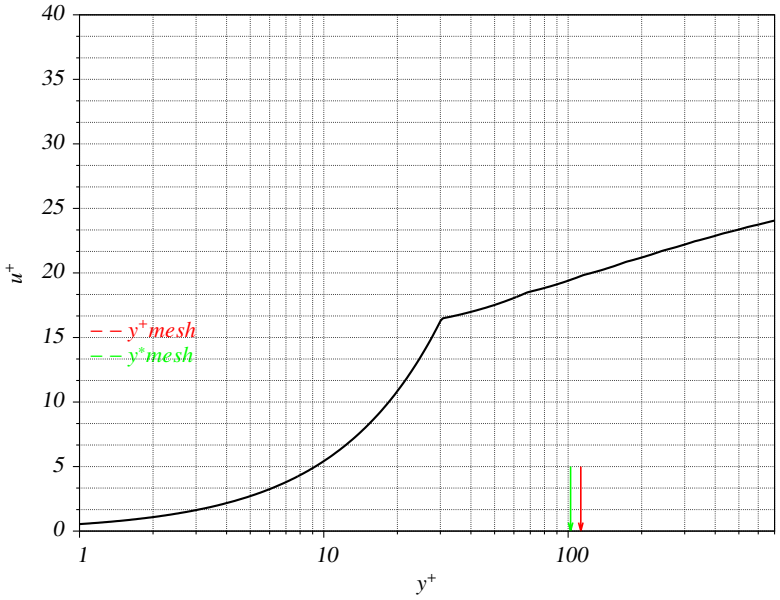
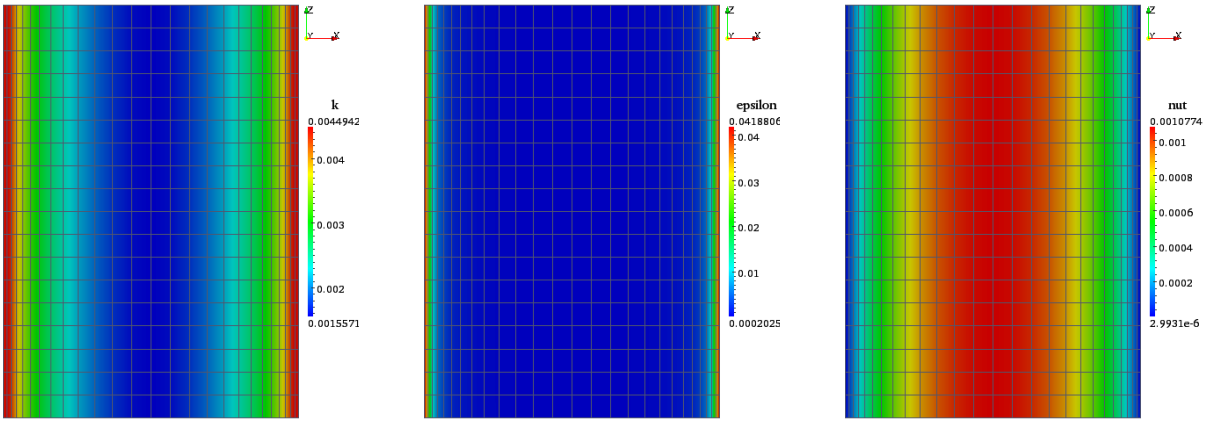


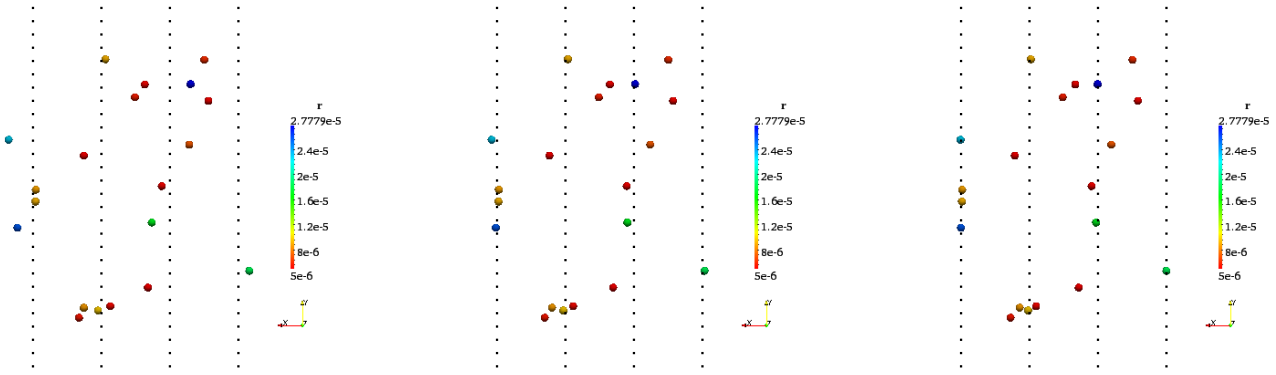
Fig. (A.18) *Log of the wall profile*



(a) *k Field* ***(b)*** *ϵ Field* ***(c)*** *μ_t Field*

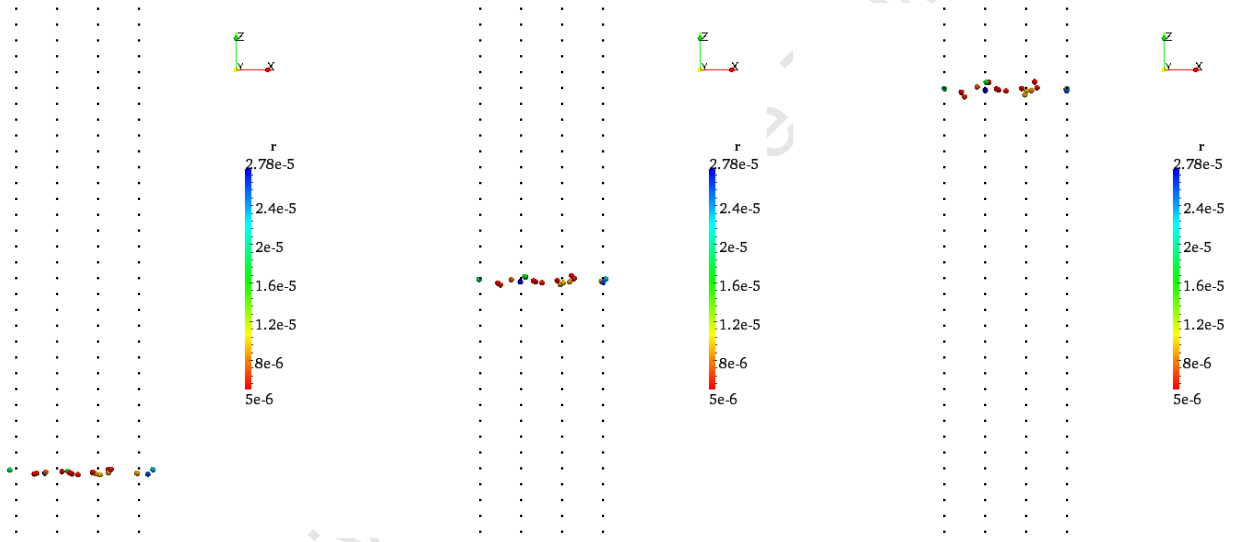
Fig. (A.19) *Kinetic energy, dissipation rate and turbulent viscosity fields, xz slice plane*

Particulate Response



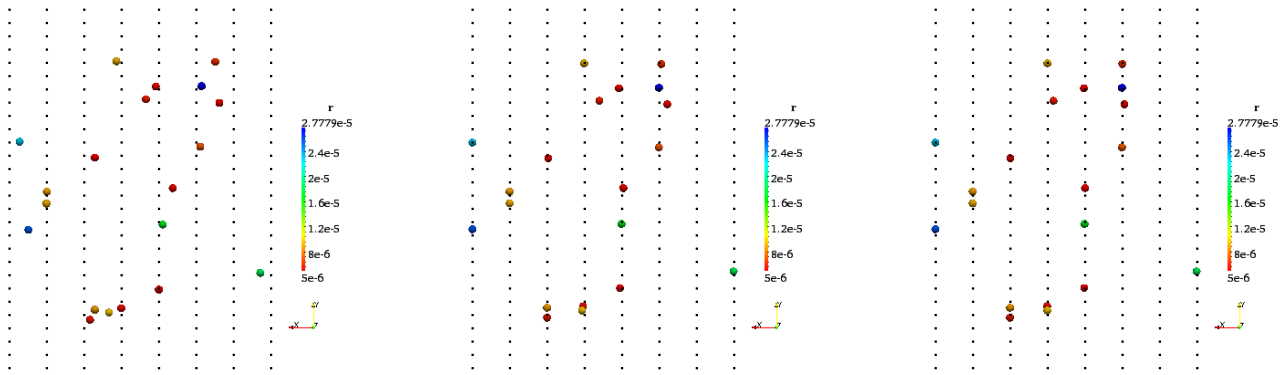
(a) t_0, *f=14794* *(b)* $t_{0.25}$, *f=14794* *(c)* $t_{0.5}$, *f=14794*

Fig. (A.20) *Case study 3,* $U_z = 1.0\ ms^{-1}$*, particle visualization xy plane, f=14794 Hz*



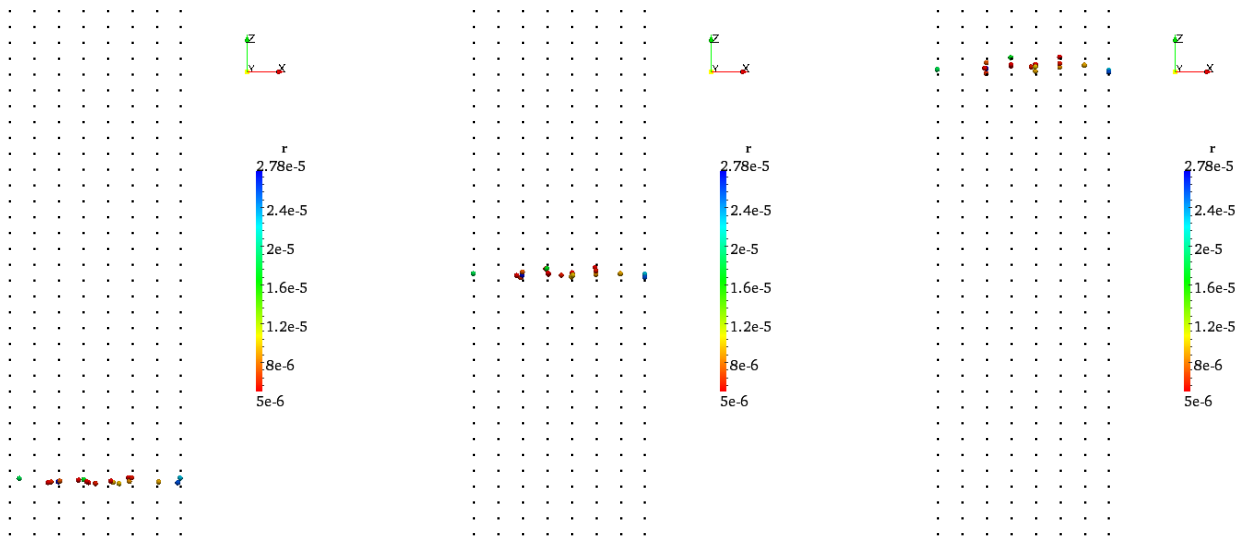
(a) t_0, *f=14794* *(b)* $t_{0.25}$, *f=14794* *(c)* $t_{0.5}$, *f=14794*

Fig. (A.21) *Case study 3,* $U_z = 1.0\ ms^{-1}$*, particle visualization xz plane, f=14794 Hz*



(a) t_0, *f=26629* *(b)* $t_{0.25}$, *f=26629* *(c)* $t_{0.5}$, *f=26629*

Fig. (A.22) *Case study 3,* $U_z = 1.0\ ms^{-1}$*, particle visualization xy plane, f=26629 Hz*



(a) t_0, *f=26629* *(b)* $t_{0.25}$, *f=26629* *(c)* $t_{0.5}$, *f=26629*

Fig. (A.23) *Case study 3,* $U_z = 1.0\ ms^{-1}$*, particle visualization xz plane, f=26629 Hz*

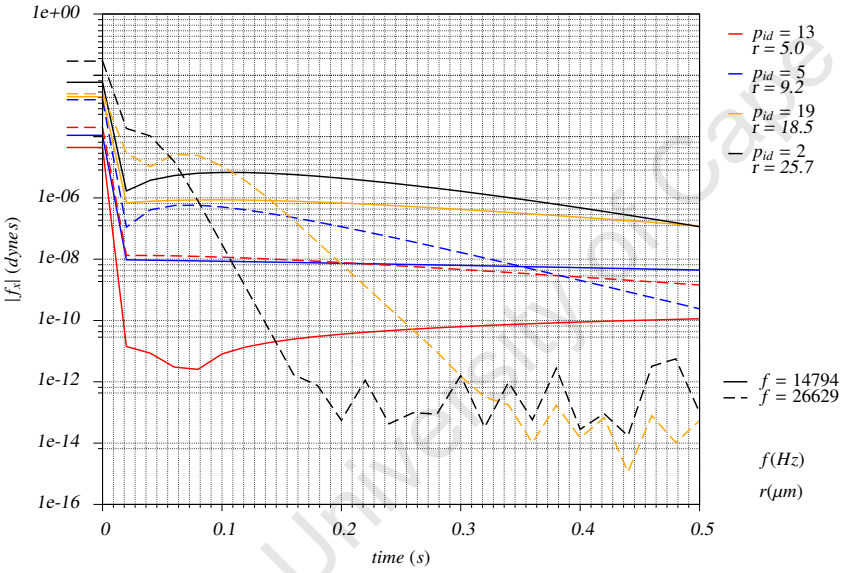


Fig. (A.24) *Primary acoustic force at two driving frequencies t ∈ [0:0.5]*

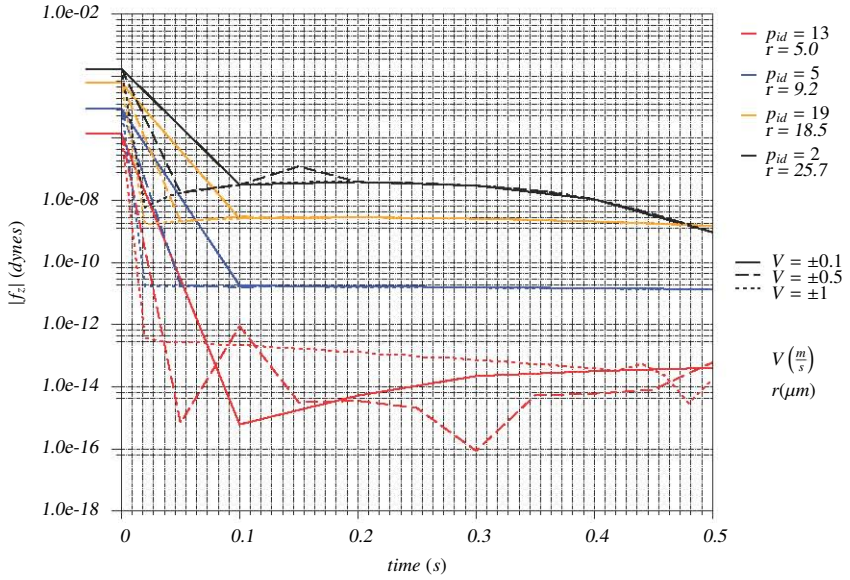


Fig. (A.25) *Drag force at various flow rates, f=14794 Hz t ∈ [0:0.5]*

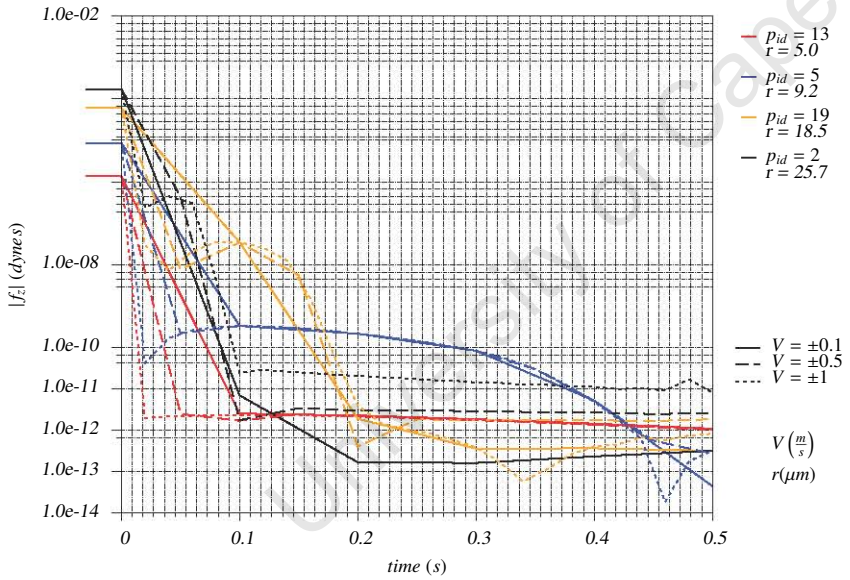


Fig. (A.26) *Drag force at various flow rates, f=26629 Hz t ∈ [0:0.5]*

APPENDIX B

TABLES

Acoustic Theory

Acoustic Fields in Fluids

Tbl. (B.1) *Fluid properties at 20 °C [50]*

Quantity	Symbol	Magnitude	Units
temperature	T	20	$^{\circ}C$
density	ρ_f	998	$\frac{kg}{m^3}$
Bulk Modulus	β_T	2.18×10^9	$\frac{N}{m^2}$
ratio of specific heats	γ	1.004	-
speed of sound	c_f	1480	$\frac{m}{s}$
coefficient of viscosity	η	0.001	$\frac{Ns}{m^2}$
characteristic impedance	$\rho_f\, c_f$	1.47×10^6	$\frac{kg}{m^2}$

DEM

Material and Physical Properties

Tbl. (B.2) *Material specific properties*

Quantity	Symbol	Units	Material 1	Material 2
			steel	silica quartz
density	ρ	$\frac{kg}{m^3}$	7900[c]	2300-2700[a]
Youngs Modulus	Y	GPa	200[c]	70[b]
poissons Ratio	ν	–	0.29[c]	0.18[b]

[a] [62]

[b] [63] Youngs Modulus Si range 130-190 GPa, Poissons Ratio range 0.064-0.36, [64] Youngs Modulus SiO_2 range 65-75 Gpa, Poissons Ratio range 0.15-0.19, [7] Youngs Modulus Fuzed Quartz range 50-70 GPa, [65] Youngs Modulus Si range 130-190 GPa, Poissons Ratio range 0.22-0.28

[c] [66] AISI 1005 Low Carbon Steel [67] Some Useful Numbers GEOL 615

Tbl. (B.3) *Contact properties*

Material 1	Material 2	μ	$C.O.R$
steel	silica quartz	0.45[a]	0.5[b]
silica quartz	steel	0.45[a]	0.5[b]
steel	steel	0.7[d]	0.7[c]
silica quartz	silica quartz	0.65[e]	0.75[f]

[a] [32] range 0.25-0.8, [26] 0.3, [18] range 0.45-0.58

[b] [32] range 0.4-0.85,[53] pg 29, range 0.3-0.6

[c] [49] range 0.5-0.8, [29] range 0.6-0.8

[d] [68]

[e] [36] range 0.6-0.85, GEOL 615 range 0.5-0.8, [2] range 0.5-0.8

[f] [2] range 0.5-1.0

DEM LIGGGHS Modifications

CGS to SI Conversion

Tbl. (B.4) *CGS $\Longleftrightarrow$ SI conversion table [69]*

Quantity	Symbol	CGS unit	Definition	Equivalent in SI units
length, position	L, x	cm	1/100 of metre	$= 10^{-2}$ m
mass	m	g	1/1000 of kilogram	$= 10^{-3}$ kg
time	t	s	1 second	= 1 s
velocity	v	cm/s	cm/s	$= 10^{-2}$ m/s
acceleration	a	Gal	cm/s^2	$= 10^{-2}$ m/s^2
force	F	dyn	gcm/s^2	$= 10^{-5}$ N
energy	E	erg	gcm^2/s^2	$= 10^{-7}$ J
power	P	erg/s	gcm^2/s^3	$= 10^{-7}$ W
pressure	p	Ba	$g/(cms^2)$	$= 10^{-1}$ Pa
dynamic viscosity	μ	P	g/(cms)	$= 10^{-1}$ Pas
kinematic viscosity	ν	St	cm^2/s	$= 10^{-4}$ m^2/s
wavenumber	k	cm^{-1}	cm^{-1}	$= 100$ m^{-1}

Model Validation

Analytic Fields

Tbl. (B.5) *Pressure nodes corresponding to various driving frequencies*

n	8	10	12	14	16	18	20
f	11835.5	14794.3	17753.2	20712.1	23670.9	26629.8	29588.6
node							
n1	0.03125	0.025	0.0208333	0.0178571	0.015625	0.0138889	0.0125
n2	0.09375	0.075	0.0625	0.0535714	0.046875	0.0416667	0.0375
n3	0.15625	0.125	0.104167	0.0892857	0.078125	0.0694444	0.0625
n4	0.21875	0.175	0.145833	0.125	0.109375	0.0972222	0.0875
n5	0.28125	0.225	0.1875	0.160714	0.140625	0.125	0.1125
n6	0.34375	0.275	0.229167	0.196429	0.171875	0.152778	0.1375
n7	0.40625	0.325	0.270833	0.232143	0.203125	0.180556	0.1625
n8	0.46875	0.375	0.3125	0.267857	0.234375	0.208333	0.1875
n9	–	0.425	0.354167	0.303571	0.265625	0.236111	0.2125
n10	–	0.475	0.395833	0.339286	0.296875	0.263889	0.2375
n11	–	–	0.4375	0.375	0.328125	0.291667	0.2625
n12	–	–	0.479167	0.410714	0.359375	0.319444	0.2875
n13	–	–	–	0.446429	0.390625	0.347222	0.3125
n14	–	–	–	0.482143	0.421875	0.375	0.3375
n15	–	–	–	–	0.453125	0.402778	0.3625
n16	–	–	–	–	0.484375	0.430556	0.3875
n17	–	–	–	–	–	0.458333	0.4125
n18	–	–	–	–	–	0.486111	0.4375
n19	–	–	–	–	–	–	0.4625
n20	–	–	–	–	–	–	0.4875

CFD Flow Development

Case Study 1 $U_{max} = 0.1ms^{-1}$

Tbl. (B.6) *Boundary layer thickness*

200[a]	U^{x}_{max}	0.102454	U^{x}_{ave}	0.0900244	U^{x99}_{ave}	0.0891242	$\delta_{99}(U^{x99}_{ave})$	0.062145
200[a]	$\bar{U}^{x}_{max}$	0.0873692	$\bar{U}^{x}_{ave}$	0.0767932	$\bar{U}^{x99}_{ave}$	0.0760253	$\delta_{99}(\bar{U}^{x99}_{ave})$	0.0268564
(19×19)[b]	U^{xy}_{max}	0.102464	U^{xy}_{ave}	0.0784596	U^{xy99}_{ave}	0.077675	$\delta_{99}(U^{xy99}_{ave})$	0.0274392
500[c]	U^{x}_{max}	0.10245	U^{x}_{ave}	0.090029	U^{x99}_{ave}	0.089129	$\delta_{99}(U^{x99}_{ave})$	0.062172
500[c]	$\bar{U}^{x}_{max}$	0.087369	$\bar{U}^{x}_{ave}$	0.076797	$\bar{U}^{x99}_{ave}$	0.076029	$\delta_{99}(\bar{U}^{x99}_{ave})$	0.026858
(101×101)[d]	U^{xy}_{max}	0.10246	U^{xy}_{ave}	0.080656	U^{xy99}_{ave}	0.07985	$\delta_{99}(U^{xy99}_{ave})$	0.031424

[a] Line Data (corresponding number = number of one dimensional data points used to integrate linear profile)
[b] Surface Node Data (corresponding product = number of data points in each dimension used to integrate surface profile)
[c] Splined Line data (corresponding number = number of one dimensional spline points used to integrate linear profile)
[d] Splined Surface Node data (corresponding product = number of spline points in each dimension used to integrate surface profile)

Case Study 2 $U_{max} = 0.5ms^{-1}$

Tbl. (B.7) *Boundary layer thickness*

200	U^{x}_{max}	0.497065	U^{x}_{ave}	0.454606	U^{x99}_{ave}	0.45006	$\delta_{99}(U^{x99}_{ave})$	0.0692607
200	$\bar{U}^{x}_{max}$	0.437086	$\bar{U}^{x}_{ave}$	0.399849	$\bar{U}^{x99}_{ave}$	0.39585	$\delta_{99}(\bar{U}^{x99}_{ave})$	0.0252492
(21×21)	U^{xy}_{max}	0.497124	U^{xy}_{ave}	0.412585	U^{xy99}_{ave}	0.40846	$\delta_{99}(U^{xy99}_{ave})$	0.0317415
500	U^{x}_{max}	0.49706	U^{x}_{ave}	0.45467	U^{x99}_{ave}	0.45013	$\delta_{99}(U^{x99}_{ave})$	0.069348
500	$\bar{U}^{x}_{max}$	0.43709	$\bar{U}^{x}_{ave}$	0.39991	$\bar{U}^{x99}_{ave}$	0.39591	$\delta_{99}(\bar{U}^{x99}_{ave})$	0.025279
(101×101)	U^{xy}_{max}	0.49712	U^{xy}_{ave}	0.4159	U^{xy99}_{ave}	0.41174	$\delta_{99}(U^{xy99}_{ave})$	0.033431

Case Study 3 $U_{max} = 1.0ms^{-1}$

Tbl. (B.8) *Boundary layer thickness*

200	U^{x}_{max}	0.997376	U^{x}_{ave}	0.921665	U^{x99}_{ave}	0.912448	$\delta_{99}(U^{x99}_{ave})$	0.0703468
200	$\bar{U}^{x}_{max}$	0.885282	$\bar{U}^{x}_{ave}$	0.818246	$\bar{U}^{x99}_{ave}$	0.810064	$\delta_{99}(\bar{U}^{x99}_{ave})$	0.0251754
(27×27)	U^{xy}_{max}	0.997466	U^{xy}_{ave}	0.845159	U^{xy99}_{ave}	0.836707	$\delta_{99}(U^{xy99}_{ave})$	0.0322063
500	U^{x}_{max}	0.99738	U^{x}_{ave}	0.9218	U^{x99}_{ave}	0.91258	$\delta_{99}(U^{x99}_{ave})$	0.070451
500	$\bar{U}^{x}_{max}$	0.88528	$\bar{U}^{x}_{ave}$	0.81837	$\bar{U}^{x99}_{ave}$	0.81019	$\delta_{99}(\bar{U}^{x99}_{ave})$	0.025207
(101×101)	U^{xy}_{max}	0.99747	U^{xy}_{ave}	0.84736	U^{xy99}_{ave}	0.83889	$\delta_{99}(U^{xy99}_{ave})$	0.032872

CFDEM Particulate Response Behaviour

Tbl. (B.9) *Particle list*

p_{id}	$r(\mu m)$	$\rho\left(\frac{kg}{m^3}\right)$
1	5.0	2372
2	25.7	2466
3	8.7	2673
4	17.1	2500
5	9.2	2687
6	5.1	2655
7	5.2	2401
8	9.8	2361
9	9.2	2679
10	5.1	2389
11	6.1	2489
12	7.2	2477
13	5.0	2625
14	5.8	2473
15	5.1	2399
16	27.7	2600
17	5.6	2635
18	8.9	2449
19	18.5	2315
20	23.0	2675

Tbl. (B.10) *Case study 1, $U_z = 0.1\ ms^{-1}$, particle related data, f=14794 Hz*

p_{id}	13	6	5	4	19	2
$r(\mu m)$	5.0	5.1	9.2	17.2	18.5	25.7
$\rho(kgm^{-3})$	2624	2655	2686	2500	2315	2465
$(x, z, t)_{in}$	(28.78, 10.80, 0.00)	(19.66, 10.27, 0.00)	(27.18, 10.31, 0.00)	(23.89, 10.57, 0.00)	(16.58, 10.75, 0.00)	(33.79, 10.26, 0.00)
$(x, z, t)_f$	(27.59, 61.85, 5.00)	(17.85, 60.12, 5.00)	(27.50, 60.02, 5.00)	(22.50, 61.33, 5.00)	(17.50, 60.07, 5.00)	(32.50, 59.81, 5.00)
$(x, z, t)_n$	(27.65, 52.74, 4.11)	–	(27.35, 14.28, 0.40)	(22.64, 15.24, 0.46)	(17.36, 13.81, 0.31)	(32.63, 12.54, 0.23)
$(\delta x, \delta z, \delta t)$	(-1.13, 41.94, 4.11)	–	(0.17, 3.98, 0.40)	(-1.25, 4.68, 0.46)	(0.78, 3.06, 0.31)	(-1.15, 2.28, 0.23)
$(y, z, t)_{in}$	(27.59, 10.80, 0.00)	(31.60, 10.27, 0.00)	(34.64, 10.31, 0.00)	(22.69, 10.57, 0.00)	(19.16, 10.75, 0.00)	(22.31, 10.26, 0.00)
$(y, z, t)_f$	(27.59, 61.85, 5.00)	(31.60, 60.12, 5.00)	(34.64, 60.02, 5.00)	(22.69, 61.33, 5.00)	(19.16, 60.07, 5.00)	(22.31, 59.81, 5.00)
$(\delta y, \delta z, \delta t)$	(0.00, 51.05, 5.00)	(0.00, 49.84, 5.00)	(0.00, 49.71, 5.00)	(0.00, 50.76, 5.00)	(0.00, 49.32, 5.00)	(0.00, 49.55, 5.00)
(fx_{lo}, fx_{hi})	(-4.4e-05, 3.1e-10)	(-1.9e-05, 3.6e-10)	(-1.1e-08, 1.1e-04)	(-1.7e-03, 6.2e-07)	(-9.0e-07, 2.0e-03)	(-5.9e-03, 7.0e-06)
(fz_{lo}, fz_{hi})	(-1.4e-06, 1.3e-13)	(-1.5e-06, 8.2e-13)	(-8.7e-06, 5.1e-12)	(-5.2e-05, 3.1e-10)	(-6.0e-05, 1.1e-10)	(-1.7e-04, 3.1e-10)
fx_{abs}	(4.4e-05)	(1.9e-05)	(1.1e-04)	(1.7e-03)	(2.0e-03)	(5.9e-03)
fz_{abs}	(1.4e-06)	(1.5e-06)	(8.7e-06)	(5.2e-05)	(6.0e-05)	(1.7e-04)
$fx_f(t)$	7.2e-11(5.000)	2.8e-10(5.000)	-5.8e-13(5.000)	-1.0e-14(5.000)	-2.5e-15(5.000)	7.7e-15(5.000)
$fz_f(t)$	-6.1e-14(5.000)	-1.4e-13(5.000)	6.8e-14(5.000)	2.8e-13(5.000)	3.3e-13(5.000)	1.1e-12(5.000)
$vz(t)$	10.19(0.010)	10.05(0.010)	9.94(0.010)	10.17(0.010)	9.73(0.010)	9.84(0.010)
$vz_f(t)$	10.24(5.000)	9.95(5.000)	9.94(5.000)	10.15(5.000)	9.86(5.000)	9.91(5.000)

p_{id} particle identifier number
$()_{in}$ initial time, t simulation time
$()_f$ final time
$()_n$ particle position and simulation time at which particle is first located within 1.5 mm of its nearest node
$()_{lo,hi}$ force bounding values, minima and maxima of the data set
$()_{abs}$ absolute maximum value of force
[a] position and time increment from initial configuration to nodal configuration
[b] particle velocity and simulation time at which particle has attained flow velocity to within 1.0 cms^{-1} tolerance
note $vz(t = 0) = 0$ for all particles
units (x,y,z)(cm) (t)(s) (fx,fz)(dynes)

Tbl. (B.11) *Case study 1, $U_z = 0.1\ ms^{-1}$, particle related data, f=26629 Hz*

p_{id}	13	6	5	4	19	2
$r(\mu m)$	5.0	5.1	9.2	17.2	18.5	25.7
$\rho(kgm^{-3})$	2624	2655	2686	2500	2315	2465
$(x, z, t)_{in}$	(28.78, 10.80, 0.00)	(19.66, 10.27, 0.00)	(27.18, 10.31, 0.00)	(23.89, 10.57, 0.00)	(16.58, 10.75, 0.00)	(33.79, 10.26, 0.00)
$(x, z, t)_f$	(29.17, 61.77, 5.00)	(20.83, 60.53, 5.00)	(26.39, 60.02, 5.00)	(23.61, 61.32, 5.00)	(15.28, 59.32, 5.00)	(34.72, 59.12, 5.00)
$(x, z, t)_n$	(29.02, 12.64, 0.18)	(20.69, 15.60, 0.53)	(26.53, 11.50, 0.12)	(23.72, 10.77, 0.02)	(15.43, 11.53, 0.08)	(34.67, 10.65, 0.04)
$(\delta x, \delta z, \delta t)$	(0.24, 1.83, 0.18)	(1.03, 5.33, 0.53)	(-0.65, 1.19, 0.12)	(-0.17, 0.20, 0.02)	(-1.15, 0.78, 0.08)	(0.88, 0.39, 0.04)
$(y, z, t)_{in}$	(27.59, 10.80, 0.00)	(31.60, 10.27, 0.00)	(34.64, 10.31, 0.00)	(22.69, 10.57, 0.00)	(19.16, 10.75, 0.00)	(22.31, 10.26, 0.00)
$(y, z, t)_f$	(27.59, 61.77, 5.00)	(31.60, 60.53, 5.00)	(34.64, 60.02, 5.00)	(22.69, 61.32, 5.00)	(19.16, 59.32, 5.00)	(22.31, 59.12, 5.00)
$(\delta y, \delta z, \delta t)$	(0.00, 50.96, 5.00)	(0.00, 50.26, 5.00)	(0.00, 49.71, 5.00)	(0.00, 50.75, 5.00)	(0.00, 48.56, 5.00)	(0.00, 48.86, 5.00)
(fx_{lo}, fx_{hi})	(-1.7e-08, 2.0e-04)	(-2.0e-08, 1.3e-04)	(-1.6e-03, 6.5e-07)	(-6.0e-03, 1.9e-05)	(-2.5e-03, 3.0e-05)	(-1.9e-04, 2.9e-02)
(fz_{lo}, fz_{hi})	(-1.4e-06, 2.9e-11)	(-1.5e-06, 2.5e-11)	(-8.7e-06, 1.1e-10)	(-5.2e-05, 1.1e-10)	(-6.0e-05, 3.5e-08)	(-1.7e-04, 3.1e-10)
fx_{abs}	(2.0e-04)	(1.3e-04)	(1.6e-03)	(6.0e-03)	(2.5e-03)	(2.9e-02)
fz_{abs}	(1.4e-06)	(1.5e-06)	(8.7e-06)	(5.2e-05)	(6.0e-05)	(1.7e-04)
$fx_f(t)$	5.8e-17(5.000)	2.0e-17(5.000)	-3.7e-16(5.000)	3.7e-17(5.000)	9.1e-16(5.000)	-3.5e-15(5.000)
$fz_f(t)$	8.4e-15(5.000)	4.0e-15(5.000)	6.8e-14(5.000)	2.9e-13(5.000)	3.3e-13(5.000)	8.1e-13(5.000)
$vz(t)$	10.19(0.010)	10.05(0.010)	9.95(0.010)	10.18(0.010)	9.74(0.010)	9.88(0.010)
$vz_f(t)$	10.19(5.000)	10.05(5.000)	9.94(5.000)	10.15(5.000)	9.71(5.000)	9.77(5.000)

Tbl. (B.12) *Case study 2, $U_z = 0.5\ ms^{-1}$, particle related data, f=14794 Hz*

p_{id}	13	6	5	4	19	2
$r(\mu m)$	5.0	5.1	9.2	17.2	18.5	25.7
$\rho(kgm^{-3})$	2624	2655	2686	2500	2315	2465
$(x, z, t)_{in}$	(28.78, 10.80, 0.00)	(19.66, 10.27, 0.00)	(27.18, 10.31, 0.00)	(23.89, 10.57, 0.00)	(16.58, 10.75, 0.00)	(33.79, 10.26, 0.00)
$(x, z, t)_f$	(28.34, 60.51, 1.00)	(19.40, 59.19, 1.00)	(27.46, 58.91, 1.00)	(22.50, 60.19, 1.00)	(17.50, 59.46, 1.00)	(32.50, 59.30, 1.00)
$(x, z, t)_n$	–	–	(27.35, 29.75, 0.40)	(22.65, 33.14, 0.46)	(17.35, 25.52, 0.30)	(32.64, 21.21, 0.23)
$(\delta x, \delta z, \delta t)$	–	–	(0.17, 19.44, 0.40)	(-1.24, 22.58, 0.46)	(0.78, 14.76, 0.30)	(-1.15, 10.95, 0.23)
(fx_{lo}, fx_{hi})	(-4.4e-05, 2.1e-10)	(-1.9e-05, -2.2e-10)	(-9.8e-09, 1.1e-04)	(-1.7e-03, 6.0e-07)	(-8.7e-07, 2.0e-03)	(-5.9e-03, 6.8e-06)
(fz_{lo}, fz_{hi})	(-1.4e-06, 1.7e-11)	(-1.5e-06, 6.3e-12)	(-8.7e-06, -8.1e-12)	(-5.2e-05, 3.8e-10)	(-6.0e-05, -3.4e-12)	(-1.7e-04, 1.2e-07)
fx_{abs}	(4.4e-05)	(1.9e-05)	(1.1e-04)	(1.7e-03)	(2.0e-03)	(5.9e-03)
fz_{abs}	(1.4e-06)	(1.5e-06)	(8.7e-06)	(5.2e-05)	(6.0e-05)	(1.7e-04)
$fx_f(t)$	2.1e-10(1.000)	-2.9e-10(1.000)	-1.7e-09(1.000)	1.0e-08(1.000)	-2.9e-09(1.000)	7.5e-11(1.000)
$fz_f(t)$	-4.4e-14(1.000)	1.6e-13(1.000)	-8.3e-12(1.000)	-6.0e-11(1.000)	-3.4e-12(1.000)	1.1e-12(1.000)
$vz(t)$	49.70(0.005)	48.92(0.005)	48.61(0.005)	49.64(0.005)	48.10(0.005)	48.50(0.005)
$vz_f(t)$	49.70(1.000)	48.92(1.000)	48.60(1.000)	49.62(1.000)	48.83(1.000)	49.14(1.000)

Tbl. (B.13) *Case study 2, $U_z = 0.5\ ms^{-1}$, particle related data, f=26629 Hz*

p_{id}	13	6	5	4	19	2
$r(\mu m)$	5.0	5.1	9.2	17.2	18.5	25.7
$\rho(kgm^{-3})$	2624	2655	2686	2500	2315	2465
$(x, z, t)_{in}$	(28.78, 10.80, 0.00)	(19.66, 10.27, 0.00)	(27.18, 10.31, 0.00)	(23.89, 10.57, 0.00)	(16.58, 10.75, 0.00)	(33.79, 10.26, 0.00)
$(x, z, t)_f$	(29.17, 60.19, 1.00)	(20.83, 59.19, 1.00)	(26.39, 58.91, 1.00)	(23.61, 60.18, 1.00)	(15.28, 58.83, 1.00)	(34.72, 58.69, 1.00)
$(x, z, t)_n$	(29.02, 19.49, 0.17)	(20.69, 36.20, 0.53)	(26.53, 16.14, 0.12)	(23.75, 11.31, 0.01)	(15.39, 14.84, 0.09)	(34.63, 11.95, 0.04)
$(\delta x, \delta z, \delta t)$	(0.24, 8.69, 0.17)	(1.03, 25.93, 0.53)	(-0.65, 5.83, 0.12)	(-0.14, 0.74, 0.01)	(-1.19, 4.08, 0.09)	(0.84, 1.69, 0.04)
(fx_{lo}, fx_{hi})	(-1.3e-08, 2.0e-04)	(-1.5e-08, 1.3e-04)	(-1.6e-03, 5.9e-07)	(-6.0e-03, 2.0e-05)	(-2.5e-03, 2.9e-05)	(-1.9e-04, 2.9e-02)
(fz_{lo}, fz_{hi})	(-1.4e-06, 2.1e-11)	(-1.5e-06, 6.5e-12)	(-8.7e-06, 5.9e-11)	(-5.2e-05, 3.4e-11)	(-6.0e-05, 3.6e-08)	(-1.7e-04, 7.2e-08)
fx_{abs}	(2.0e-04)	(1.3e-04)	(1.6e-03)	(6.0e-03)	(2.5e-03)	(2.9e-02)
fz_{abs}	(1.4e-06)	(1.5e-06)	(8.7e-06)	(5.2e-05)	(6.0e-05)	(1.7e-04)
$fx_f(t)$	-6.5e-11(1.000)	-6.1e-10(1.000)	4.8e-15(1.000)	-4.2e-14(1.000)	-2.9e-13(1.000)	1.2e-12(1.000)
$fz_f(t)$	-4.4e-14(1.000)	-8.2e-13(1.000)	1.6e-13(1.000)	1.5e-12(1.000)	1.5e-12(1.000)	2.2e-12(1.000)
$vz(t)$	49.70(0.005)	48.92(0.005)	48.61(0.005)	49.65(0.005)	48.10(0.005)	48.54(0.005)
$vz_f(t)$	49.33(1.000)	48.92(1.000)	48.60(1.000)	49.62(1.000)	48.08(1.000)	48.43(1.000)

Tbl. (B.14) *Case study 3, $U_z = 1.0\ ms^{-1}$, particle related data, f=14794 Hz*

p_{id}	13	6	5	4	19	2
$r(\mu m)$	5.0	5.1	9.2	17.2	18.5	25.7
$\rho(kgm^{-3})$	2624	2655	2686	2500	2315	2465
$(x, z, t)_{in}$	(28.78, 10.80, 0.00)	(19.66, 10.27, 0.00)	(27.18, 10.31, 0.00)	(23.89, 10.57, 0.00)	(16.58, 10.75, 0.00)	(33.79, 10.26, 0.00)
$(x, z, t)_f$	(28.56, 60.43, 0.50)	(19.55, 59.16, 0.50)	(27.38, 58.83, 0.50)	(22.61, 60.39, 0.50)	(17.46, 59.60, 0.50)	(32.50, 59.34, 0.50)
$(x, z, t)_n$	–	–	(27.35, 48.74, 0.40)	(22.65, 55.61, 0.45)	(17.35, 40.25, 0.30)	(32.65, 32.05, 0.22)
$(\delta x, \delta z, \delta t)$	–	–	(0.17, 38.43, 0.40)	(-1.24, 45.04, 0.45)	(0.77, 29.50, 0.30)	(-1.14, 21.79, 0.22)
(fx_{lo}, fx_{hi})	(-4.4e-05, 1.1e-10)	(-1.9e-05, -2.2e-10)	(-9.8e-09, 1.1e-04)	(-1.7e-03, 6.0e-07)	(-8.7e-07, 2.0e-03)	(-5.9e-03, 6.8e-06)
(fz_{lo}, fz_{hi})	(-1.4e-06, 1.1e-10)	(-1.5e-06, 8.2e-10)	(-8.7e-06, 2.7e-10)	(-5.2e-05, 2.0e-09)	(-6.0e-05, 3.0e-09)	(-1.7e-04, 6.7e-06)
fx_{abs}	(4.4e-05)	(1.9e-05)	(1.1e-04)	(1.7e-03)	(2.0e-03)	(5.9e-03)
fz_{abs}	(1.4e-06)	(1.5e-06)	(8.7e-06)	(5.2e-05)	(6.0e-05)	(1.7e-04)
$fx_f(t)$	1.1e-10(0.500)	-2.6e-10(0.500)	-4.4e-09(0.500)	2.2e-07(0.500)	-1.2e-07(0.500)	1.2e-07(0.500)
$fz_f(t)$	1.8e-14(0.500)	1.7e-13(0.500)	-1.3e-11(0.500)	-1.6e-09(0.500)	-1.5e-09(0.500)	-9.4e-10(0.500)
$vz(t)$	99.26(0.002)	97.78(0.002)	97.06(0.002)	99.67(0.002)	97.72(0.002)	98.22(0.002)
$vz_f(t)$	99.26(0.500)	97.78(0.500)	97.06(0.500)	99.66(0.500)	97.70(0.500)	98.16(0.500)

Tbl. (B.15) *Case study 3, $U_z = 1.0\ ms^{-1}$, particle related data, f=26629 Hz*

p_{id}	13	6	5	4	19	2
$r(\mu m)$	5.0	5.1	9.2	17.2	18.5	25.7
$\rho(kgm^{-3})$	2624	2655	2686	2500	2315	2465
$(x, z, t)_{in}$	(28.78, 10.80, 0.00)	(19.66, 10.27, 0.00)	(27.18, 10.31, 0.00)	(23.89, 10.57, 0.00)	(16.58, 10.75, 0.00)	(33.79, 10.26, 0.00)
$(x, z, t)_f$	(29.15, 60.43, 0.50)	(20.66, 59.16, 0.50)	(26.39, 58.83, 0.50)	(23.61, 60.39, 0.50)	(15.28, 59.04, 0.50)	(34.72, 58.73, 0.50)
$(x, z, t)_n$	(29.02, 27.87, 0.17)	–	(26.54, 21.76, 0.12)	(23.76, 11.95, 0.01)	(15.41, 18.76, 0.08)	(34.59, 13.39, 0.03)
$(\delta x, \delta z, \delta t)$	(0.23, 17.07, 0.17)	–	(-0.65, 11.45, 0.12)	(-0.13, 1.39, 0.01)	(-1.17, 8.00, 0.08)	(0.80, 3.13, 0.03)
(fx_{lo}, fx_{hi})	(-1.3e-08, 2.0e-04)	(-1.5e-08, 1.3e-04)	(-1.6e-03, 5.9e-07)	(-6.0e-03, 2.1e-05)	(-2.5e-03, 2.9e-05)	(-2.0e-04, 2.9e-02)
(fz_{lo}, fz_{hi})	(-1.4e-06, 3.3e-11)	(-1.5e-06, 1.0e-09)	(-8.7e-06, 8.0e-11)	(-5.2e-05, 3.9e-11)	(-6.0e-05, 3.9e-08)	(-4.2e-04, 5.5e-07)
fx_{abs}	(2.0e-04)	(1.3e-04)	(1.6e-03)	(6.0e-03)	(2.5e-03)	(2.9e-02)
fz_{abs}	(1.4e-06)	(1.5e-06)	(8.7e-06)	(5.2e-05)	(6.0e-05)	(4.2e-04)
$fx_f(t)$	-1.4e-09(0.500)	-1.2e-08(0.500)	2.4e-10(0.500)	-8.8e-15(0.500)	-5.4e-14(0.500)	-9.9e-14(0.500)
$fz_f(t)$	-1.0e-12(0.500)	-3.0e-12(0.500)	3.3e-13(0.500)	1.3e-12(0.500)	8.3e-13(0.500)	8.2e-12(0.500)
$vz(t)$	99.26(0.002)	97.78(0.002)	97.06(0.002)	99.68(0.002)	97.72(0.002)	98.27(0.002)
$vz_f(t)$	99.26(0.500)	97.78(0.500)	97.05(0.500)	99.65(0.500)	96.36(0.500)	96.88(0.500)

APPENDIX C

CODE FRAGMENTS

Modelling Theory

DEM

```
//constant time step
dt = ...;

//time steps for position and velocity update
dtv = dt;
dtf = 0.5 * dt;

time_loop
{
    for(all particles i)
    {
        dtfm= dtf / mass[i];

        //update position, update velocity with half step
        v[i] += dtfm * f[i];
        x[i] += dtv * v[i];

        //calculate forces on particles
        f[i] = ...;

        //update velocity with remaining half step
        v[i] += dtfm * f[i];
    }
}
```

Code Fragment (C.1) *LAMMPS implementation of Velocity Verlet integration*

CFD Model

Mesh

```
...
    {
        name WALL;

        dictionary
        {
            type wall;
        }
        constructFrom patches;

        patches (SIDES);
    }
    {
        name INLET;
        dictionary

        {
            type patch;
        }
        constructFrom patches;

        patches (ductinlet);
    }
    {
        name OUTLET;

        dictionary
        {
            type patch;
        }
        constructFrom patches;

        patches (ductoutlet);
    }
```

Code Fragment (C.2) *createPatchDict*

```
echo "generating .msh file"
gmsh -3 duct.geo

echo "writing mesh to foam"
gmshToFoam duct.msh

echo "patching mesh"
createPatch

# file transfer and cleaning
```

Code Fragment (C.3) *mesh.sh*

```
// defining points

Point(1) = {0, 0, 0, lc};
Point(2) = {0.5, 0, 0, lc};
Point(3) = {0, 0, 0.7, lc};
...
//defining lines from points
```

```
Line(1) = {2, 1};
Line(2) = {1, 3};
Line(3) = {3, 4};
...
//defining surfaces from lines

Line Loop(6) = {3, 4, 1, 2};
Ruled Surface(6) = {6};
Line Loop(15) = {3, 14, -8, -13};
Ruled Surface(15) = {15};
...
//defining volume from surfaces

Surface Loop(1) = {6, 28, 15, 19, 23, 27};
Volume(1) = {1};

//discretizing lines

Transfinite Line {10, 8, 3, 1, 18, 14, 13, 22} = 11 Using Bump 0.5;
Transfinite Line {9, 11, 2, 4} = 15;

//discretizing surfaces

Transfinite Surface {28,15,6,23,27,19};
Recombine Surface {28,15,6,23,27,19};

//discretizing volume

Transfinite Volume {1};

//naming surfaces and volumes

Physical Surface("ductoutlet") = {23};
Physical Surface("ductinlet") = {15};
Physical Surface("SIDES") = {28, 27, 6, 19};
Physical Volume("InternalField") = {1};
```

Code Fragment (C.4) *Gmsh input script*

Solvers and Models

```
fvVectorMatrix UEqn
// LHS of the NSE
   ∂Ū/∂t + ∇·(Ū⊗Ū) + ∇·R − ∇·2νD̄
(
     fvm::ddt(U)
   + fvm::div(phi, U)
   + turbulence->divDevReff(U)
);

UEqn.relax();

if (momentumPredictor)
// momentum predictor LHS = RHS
   ∂Ū/∂t + ∇·(Ū⊗Ū) + ∇·R − ∇·2νD̄ = −∇p̄
{
```

```
   solve(UEqn == -fvc::grad(p));
}
```

Code Fragment (C.5) *Navier Stokes Equations*

```
for (int corr = 0; corr < nCorr; corr++)
{
    // diagonal matrix UEqn.A    A

    // off diagonal H-operator matrix UEqn.H    H

    volScalarField rUA = 1.0/UEqn.A();

    U = rUA * UEqn.H();

    phi = (fvc::interpolate(U) & mesh.Sf()) + fvc::ddtPhiCorr(rUA, U, phi);

    adjustPhi(phi, U, p);

    for (int nonOrth = 0; nonOrth <= nNonOrthCorr; nonOrth++)
    {
        // Pressure Correction Equation
           ∇ · (A^-1 ∇ p) = ∇ · [(A^-1 H)·S]

        fvScalarMatrix pEqn
        (
           fvm::laplacian(rUA, p) == fvc::div(phi)
        );
        ...
        // Flux Corrector Equation
           φ = (A^-1 H)·S − (A^-1 ∇ p)

        if (nonOrth == nNonOrthCorr)
        {
           phi -= pEqn.flux();
        }
    }

  #include "continuityErrs.H"

 // Momentum Corrector Equation
    u = A^-1 H − A^-1 ∇p

  U -= rUA * fvc::grad(p);
  ...
}
```

Code Fragment (C.6) *PISO predictor corrector*

```
if (adjustTimeStep)
// adjustTimeStep flag from controlDict
{
    scalar maxDeltaTFact = maxCo/(CoNum + SMALL);
    // maxCo from controlDict
    // CoNum from CourantNo.H
```

```
    scalar deltaTFact = min(min(maxDeltaTFact, 1.0 + 0.1 * maxDeltaTFact),
    1.2);

    runTime.setDeltaT
    (
        min
        (
            deltaTFact * runTime.deltaT().value(),
            maxDeltaT
            // maxDeltaT from controlDict
        )
    );

    Info<< "deltaT = " <<  runTime.deltaT().value() << endl;
}
```

Code Fragment (C.7) *setDeltaT*

The modified **setDeltaT.H** file is included in the standard incompressible PISO algorithm as in fragment *C.8*, at the beginning of the run time loop. An additional header file **readTimeControls.H** had to be included before **setDeltaT.H**, the inclusion of which serves to reads in values **maxCo**, **adjustTimeStep** and **maxDeltaT** from the **controlDict** prior to calling **setDeltaT.H**.

A separate folder was created to contain the modified solver **pisoFoamATS** and all associated files carried through from the standard PISO solver as well as the additional **setDeltaT.H** file. The **make** file wherein the solver name is defined needed to be modified to correspond with the new source file and solver names. The three step build procedure in terminal, *rmdepall, clean, make* generates all necessary links and is done via the terminal window in the corresponding solver path directory. The procedure builds the code on top of the already existing primary and generates an executable. The advantage of **make** files is that the entire code does not need to be rebuilt for every modification. The coupling solver CFDEM and all its complimentary files are built on top of OpenFOAM in much the same way.

```
while (runTime.loop())
 {
     #include "readPISOControls.H"
     #include "readTimeControls.H"       // readTimeControls.H added

     #include "CourantNo.H"
     #include "setDeltaT.H"              // setDeltaT.H added
 ...
```

Code Fragment (C.8) *PISO header file inclusion*

Discretization Schemes

```
ddtSchemes
{
```

```
    default         Euler;
}
gradSchemes         // template Gauss <interpolationScheme>
{
    default         Gauss linear;
    grad(p)         Gauss linear;
    grad(U)         Gauss linear;
}
divSchemes          // template Gauss <interpolationScheme>
{
    default                         none;
    div(phi,U)                      Gauss limitedLinearV 1; // V appended for
    Vector
    div(phi,k)                      Gauss limitedLinear 1;
    div(phi,epsilon)                Gauss limitedLinear 1;
    div(phi,R)                      Gauss limitedLinear 1;
    div(R)                          Gauss linear;
    div(phi,nuTilda)                Gauss limitedLinear 1;
    div((nuEff*dev(grad(U).T()))    Gauss linear;
}
laplacianSchemes    // template Gauss <interpolationScheme> <snGradScheme>
{
    default                         none;
    laplacian(nuEff,U)              Gauss linear corrected;
    laplacian((1|A(U)),p)           Gauss linear corrected;
    laplacian(DkEff,k)              Gauss linear corrected;
    laplacian(DepsilonEff,epsilon)  Gauss linear corrected;
    laplacian(DREff,R)              Gauss linear corrected;
    laplacian(DnuTildaEff,nuTilda)  Gauss linear corrected;
}
interpolationSchemes
{
    default         linear;
    interpolate(U)  linear;
}
snGradSchemes       // surface normal gradient scheme
{
    default         corrected;
}
fluxRequired
{
    default         no;
    p               ;
}
```

Code Fragment (C.9) *CFD fvSchemes*

```
solvers
{
    p
    {
        solver          PCG;
        preconditioner  DIC;
        tolerance       1e-06;
        relTol          0.1;
    }
    pFinal                          // same as p
    ...
    U
    {
```

```
        solver          PBiCG;                                                  14
        preconditioner  DILU;                                                   15
        tolerance       1e-05;                                                  16
        relTol          0;                                                      17
    }                                                                           18
    k, epsilon, R, nuTilda          // same as U                                19
    ...                                                                         20
}                                                                               21
PISO                                                                            22
{                                                                               23
    nCorrectors     2;              // velocity corrections                     24
    nNonOrthogonalCorrectors 0;     // pressure corrections per velocity correction 25
    pRefCell        0;                                                          26
    pRefValue       0;                                                          27
}                                                                               28
```

Code Fragment (C.10) *fvSolution*

Boundary Conditions

```
// yPlusRAS function computes y star value                                      1
   y* = C_μ^0.25 k^0.5 y_c1 ν^-1                                                2
                                                                                3
Yp = pow(Cmu, 0.25) * sqrt(k()().boundaryField()[patchNo].patchInternalField()  4
    ) * y_[patchNo] / nu().boundaryField()[patchNo];
```

Code Fragment (C.11) *yPlusRAS*

```
// user defined function yPlusTime                                              1
// extracts wall velocity gradient                                              2
   du/dy|y=0                                                                    3
                                                                                4
gradU = stringToDouble(line);                                                   5
...                                                                             6
// extracts value of kinetic turbulent energy at the wall                       7
   k_w                                                                          8
                                                                                9
kWall = stringToDouble(line);                                                   10
...                                                                             11
// extracts value of turbulent viscosity at the wall                            12
   ν_t,w                                                                        13
                                                                                14
nutW = stringToDouble(line);                                                    15
...                                                                             16
                                                                                17
double nu = 1e-6;                                                               18
                                                                                19
// effective kinematic viscosity at the wall                                    20
   ν_e,w = ν_t,w + ν                                                            21
                                                                                22
double nuEff = nutW + nu;                                                       23
                                                                                24
// computes yPlus according to theory                                           25
   y+ = (ν_e,w du/dy|y=0)^0.5 y_c1 ν^-1                                         26
```

```
double yPlus = (1/nu) * (sqrt(nuEff * gradU)) * minx;

// computes yStar according to theory
   y* = C_mu^0.25 k^0.5 y_c1 nu^-1
double yStar = (1/nu) * (sqrt(kWall) * 0.5477) * minx;
```

Code Fragment (C.12) y^+ y^*

Two methods of simulation were proposed, one implementing cyclic z boundaries achieving seamless flow throughout the domain but producing poor pressure convergence behaviour; the other implementing pressure driven inlet outlet flow with immediate pressure convergence but incurring inaccuracies at the walls in the immediate vicinity of the inlet and outlet patches. The flow would be pressure driven in both cases. The immediate difficulty in using cyclic boundaries was that pressure could not be separately specified on the two cyclic boundaries and would require code modification to introduce a constant momentum drive source term on the RHS of the NSE, which would simulate the effect of a constant pressure gradient. The inlet outlet pressure gradient approach was adopted in this study because of its relative ease of implementation compared with that of coding a pressure drive term in the NSE. Near wall inaccuracies at inlet outlet boundaries would not affect the bulk flow in which the particles would be inserted. In addition this type of inlet outlet pressure gradient flow is commonly implemented and is hence a more reliable and tested option. The purpose of this section is to describe the procedure and setup implemented in the study to develop a converged flow profile which was to be mapped into a coupled CFD DEM framework. Since the respective CFD DEM simulation time scales were considerably different only a short simulation time would be required to produce the necessary particle response data. The same procedural setup described in this section was used to produce a number of different characteristic flow velocities.

Files for Flow Development

'0' Directory

Initialization is necessary to initiate any iterative procedure, by supplying the initial guess from which the solution is incrementally advanced. Certain field variables require explicit initialization. Variables which do not require explicit initialization are derived from those which have been specified. The velocity **U**, pressure **p**, turbulent kinetic energy **k** and dissipation rate **epsilon** have to be specified for RANS $k-\epsilon$ simulation. Turbulent kinetic energy must be initialized with a non zero value. The initial values of **k** and **epsilon** are predetermined according to equations *2.2.5 a* and *b*, and corresponding to the required flow velocity. Turbulent viscosity is computed internally using the specified **k** and **epsilon** fields as in equation *2.2.5 c*. The pressure gradient was adjusted iteratively to obtain the desired flow velocity, fragments *C.13* and *C.14*. The boundary names defined in the **createPatchDict** are referenced in fragments *C.15* and *C.16*. The values used in the corresponding script fragments were based on producing a maximum flow velocity of 0.5 ms^{-1} at the centreline of the duct.

```
...
dimensions    [0 1 -1 0 0 0 0];          // flow velocity units

internalField   uniform (0 0 0);         // vector field initialization
...
    WALL
    {
        type            fixedValue;
        value           uniform (0 0 0);         // no slip condition
    }
// INLET, OUTLET set to zeroGradient
```

Code Fragment (C.13) *Field variable U*

```
...
dimensions    [0 2 -2 0 0 0 0];       // kinematic pressure units

internalField   uniform 0;            // scalar field initialization
...
    WALL
    {
        type            zeroGradient;
    }
    INLET
    {
        type            fixedValue;
        value           uniform 0.0001;     // high pressure at inlet
    }
    OUTLET
    {
        type            fixedValue;
        value           uniform 0;          // low pressure at outlet
    }
```

Code Fragment (C.14) *Field variable p*

```
...
dimensions    [0 2 -2 0 0 0 0];       // kinematic turbulent energy units

internalField   uniform 4.29e-4;      // scalar field initialization (non-zero)

boundaryField
{
    WALL
    {
        type            kqRWallFunction;    // wall function
        value           uniform 4.29e-4;    // k value at the wall
    }
    INLET
    {
        type            zeroGradient;
    }
    OUTLET
    {
        type            zeroGradient;
    }
```

```
}
```

Code Fragment (C.15) *Field variable k*

```
...
dimensions    [0 2 -3 0 0 0 0];       // dissipation rate of kinematic turbulent
    energy

internalField   uniform 6.11e-07;    // scalar field initialization
...
    WALL
    {
        type            epsilonWallFunction;     // wall function
        value           uniform 6.11e-07;        // epsilon value at the wall
    }
// INLET, OUTLET set to zeroGradient            // same as in k
}
```

Code Fragment (C.16) *Field variable epsilon*

'constant' Directory

Turbulence properties are specified in the **turbulenceProperties** and **RASProperties** files in the **constant** directory. The **turbulenceProperties** file, not documented contains the keyword **RASModel** and the **RASProperties** file contains the keyword **kEpsilon** as in fragment *C.17*. **kEpsilon** specifies the turbulence sub-model used for the simulation. The kinematic laminar viscosity is specified in the **transportProperties** file also located in the **constant** directory. The files may contain additional entries if more complex models are used requiring coefficients and other supplements.

```
}
RASModel        kEpsilon;

turbulence      on;

printCoeffs     on;
...
```

Code Fragment (C.17) *RASProperties*

'system' Directory

The **sampleDict** file is used to specify all lines and surfaces which are to be populated with specified field variable data for plotting purposes. The **sampleDict** is invoked by a calling function **sample**. The sampling function extracts data only from already existing field variable files. The wall velocity gradient field is not produced by default during simulation, hence a **wallGradU** file, written by calling **wallGradU** must exist prior to sampling. The breakdown of the sampling function is detailed

in fragment *C.18*. Four sample lines are defined, the last of which is a line segment, line *40*. The line segment is comprised of points generated by an external user defined function with a specific spacing ratio between successive sampling points. The external code produces closely spaced points near the wall which progressively become farther spaced moving away from the wall of the geometry. This allows for an adequate near wall velocity gradient resolution which is useful for post processing purposes. Another two line segments crossing the duct square section are used to sample specified field variables in the xz and yz planes particularly useful for extracting velocity flow profiles. The remaining line segment is used to sample all field variables along the height of the control volume useful in monitoring pressure gradient during simulation. Line discretization may also be specified alongside any **nPoints** keyword which allows the degree of resolution to be adjusted. A cross sectional surface area at the mid plane of the simulation domain $z = 0.35\ m$ is also sampled with respect to all field variables marked in the **sampleDict**. It should be noted that surfaces cannot be discretized in the same way as lines can. The output resolution is mesh dependent and sample points correspond with nodal vertices. Data files are produced at write intervals which were used for plotting purposes and to extract specific data entries or sets for use in subsequent calculations. Matrix type output was produced, specified by the **raw** format keyword lines *1* and *3* in fragment *C.18*.

In addition to sampling lines and surfaces, probe points within the control volume were specified in the **controlDict** which were used to monitor variable evolution in simulation time. Probe output is written to a **probes** folder in the case directory. The probe function is defined in fragment *C.19* lines *24* to *40*. Probe points are read from an external file line *33* which allows a large number of algorithmically generated probe points to be activated, however only three probes were used in this study at $(x = 0.25, y = 0.25, z = 0.05)$, $(x = 0.25, y = 0.25, z = 0.35)$ and $(x = 0.25, y = 0.25, z = 0.65)$. The probe sampling time increment may be specified, usually smaller than the write interval, in which case time related data is accumulated and written to file along with all other output at write interval.

The main purpose of the **controlDict** is to define simulation run parameters and settings. Settings lines *2* to *6* allow consecutive calls to be made to the OpenFOAM solver each advancing the simulation for a number of time steps from a previously written time, specified in line *10* i.e. incrementally progressing the simulation via successive calls to run the solver. This incremental method of advancing the solution allows intermediary functions to be called between runs to produce effectively run time statistics. Lines *16* to *20* were required to address ATS parameters required by the modified PISO solver detailed in section *3.2.2*.

```
setFormat raw;          // output format x/y/z fields

surfaceFormat raw;      // output format (x,y,z) fields

fields                  // field data to be sampled
(
    U
    p
    k
    nut
    wallGradU
```

```
);

sets    // name of folder containing line data
(
    lineX1
    {
        type        uniform;
        axis        x;
        start       (0 0.25 0.35);
        end         (0.5 0.25 0.35);
        nPoints     200;
    }
    lineZ1
    {
        type        uniform;
        axis        z;
        start       (0.25 0.25 0);
        end         (0.25 0.25 0.7);
        nPoints     200;
    }
    lineY1
    {
        type         uniform;
        axis         y;
        start        (0.25 0 0.35);
        end          (0.25 0.5 0.35);
        nPoints      200;
    }
    lineUlogY
    {
        type        cloud;
        axis        xyz;
        points      (
                    #include "UlogYProbePoints"
                    // externally generated points
                    );
    }
);

surfaces    // name of folder containing plane data
(
    constantPlane
    {
        type            plane;
        basePoint       (0.25 0.25 0.35);
        normalVector    (0 0 1);
        interpolate     true;
    }
);
```

Code Fragment (C.18) *sampleDict*

```
...
startFrom       latestTime;

startTime       startFrom;

stopAt          nextWrite;

...
```

```
writeInterval   100;

...

runTimeModifiable yes;

adjustTimeStep  yes;

maxCo           0.95;

maxDeltaT       1;

functions
{
    probes
    {
        type                probes;
        functionObjectLibs ("libsampling.so");
        enabled             true;
        outputControl       timeStep;
        outputInterval      10;
        probeLocations
        (
        #include "probePoints"
        );
        fields
        (
            U
            p
        );
    }
    ...
    // averaging function specification for p and U fields
}
```

Code Fragment (C.19) *controlDict*

cfdRun Script

The **cfdRun.sh** script is the main script which is called initially from terminal and spawns all subsequent processes related to CFD flow development. The primary functions of the script are documented in fragments *C.20* to *C.22*. The **cfdRun.sh** script is used to call OpenFOAM repeatedly and perform sampling, manipulation and plotting commands at interval between successive calls. OpenFOAM is set up in such a way so as to simulate from the latest available time folder in the case directory and stop after a specified write interval. The most relevant run initializers are found in fragment *C.20* lines *2* to *6*. The write interval specified must correspond with that in the **controlDict**. **nSteps** is used to specify the total run time of the simulation overriding that set in the **controlDict**. **logEvery** is used to specify how often plots are generated and raised to screen. By setting the **logEvery** run variable appropriately plotting can be omitted entirely or set at regular interval. Regular plotting is useful when certain time dependent variable behaviour is of interest. Evolving plots can provide physical insight. Upon completion of a single run the time directories which have been written are sampled, line *23*, and data which has been written to **sets** and **surfaces** folders is renamed and copied to the

logs folder in which all plotting operations are conducted. The **yPlusTime** executable is called, line *27* which computes the current y^+ and y^* values corresponding to the wall at the height-midpoint of the simulation domain. This output is appended to a **yPlusLog** file which tracks these values for the duration of the run. The user is updated at every operation change so that bottlenecks and errors can be identified.

Calling OpenFoam with **| tee log** requests an additional log file be produced containing residual data corresponding to that run. Log data is not accumulated through successive calls to run OpenFOAM and thus requires special treatment to combine successive logs into a single file for plotting purposes. The appending operation is omitted in code fragments. A similar appending procedure is implemented to deal with run specific probe data.

The **surfacePlot** executable is called, fragment *C.21* line *3* to reformat the sampled data obtained at the cross sectional plane specified in **sampleDict**, for plotting purposes. At the user specified log interval all plotting functions are called. Plots are produced via an external plotter **gnuplot** hence the need for data formatting. All plots are commented in fragment *C.21*. Residual and probe velocity plots are raised for run time viewing as they are time based. All other plots are generated separately in the **logs** folder. Numerous plots are produced and uniquely named according to the variables in question and the time step at which they were produced. The **deltaT** executable line *22* is then called to transfer the last logged time step size to a new **controlDict** which serves to initialize a subsequent run. This produces time step continuity under the ATS scheme.

At the end of the simulation run additional plots may be produced, which extract profiles at various user selected times and generate overlapping plots on a single set of axes. The user may select an initial time, a time increment and an end time, fragment *C.22* line *3*, to specify which simulation time profiles are to be extracted and plotted on the available set of axes. These plots are raised to display as soon as they are generated and stored under unique name identifiers in the **logs** folder for subsequent access. The developed flow field is then mapped, line *17*, to the coupled CFDEM case directory at the time specified in its system **controlDict**. Mapping is followed by a call to the OpenFOAM **yPlusRAS** utility, line *19* to produce y^* statistics of the entire run. Since the function reads the available time folders from which relevant data is extracted to compute y^*, only output at write intervals can be produced. The y^* evolution data generated by the in-house function may be compared against that computed by the user defined **yPlusTime** function. The **cfdRun.sh** script is completed by a cleaning operation which may be flagged at the beginning of the script along with the other initializers.

Case file cleaning is further prompted so that the user may choose whether to keep simulation data or delete it. **paraFoam** may be run from terminal in the case directory to view results. Cleaning from the **cfdRun.sh** script is partial and allows the user to review plotted data in the **logs** folder post cleaning. Additional cleaning is done at the beginning of the **cfdRun.sh** script to ensure that any interfering files are removed before a new simulation is initiated. A separate cleaning script **clean.sh** is also included in the run directory which may be used to clean case files which had been retained. The basic structure of the **cfdRun.sh** script may be easily adapted for various solvers including LES and laminar and for different geometries, modified accordingly in terms of sampling and plotting.

```
...
writeInterval = 200
cleancase = "t"
logEvery = 1
logInterval = $((logEvery * writeInterval))
nSteps = 8

# initializations, file removal, temporary file creation
...
for (( i = 1; i < $nSteps ; i++ ))
  do

    time=$((i * writeInterval))
    export t=$time

    echo "running FOAM for $writeInterval timesteps"
    echo "current timestep: $((time-writeInterval))"
    echo "running till time: $time"
    pisoFoamATS | tee log
    ...

    wallGradU
    sample
    ...

    echo "calculating y+ and y* for current time step"
    ./yPlusTime
```

Code Fragment (C.20) *Run initialization*

```
    ...
    echo "reorganizing raw data for surface plot"
    ./surfacePlot

    if [ $(( $time % $logInterval )) -eq 0 ]
    then
        ...
        gnuplot pzTime      # linear pressure profile along z line
        gnuplot uyTime      # linear velocity profile along y line
        gnuplot uxTime      # linear velocity profile along x line
        gnuplot uxyTime     # planar velocity profile across xy plane
        gnuplot uxycoTime   # planar velocity contours across xy plane
        gnuplot ulogyTime   # linear dimensionless velocity profile along x line
        ...
        gnuplot -persist cPlotRvsT  # residual plot with time
        ...
        gnuplot cPlotUvsT   # probe point velocities with time
        ...
    fi

    echo "extracting current deltaT and updating the controlDict"
    ./deltaT
    ...
done
```

Code Fragment (C.21) *Runtime plotting*

```
...
echo -e "enter start time end time and time increment"
read  start end increment

echo "$start $end $increment"

echo "starttime=$start" >> header
echo "endtime=$end" >> header
echo "increment=$increment" >> header
...
gnuplot -persist uxTimes    # linear velocity profiles at various times along x line
gnuplot -persist uyTimes    # linear velocity profiles at various times along y line
gnuplot -persist pzTimes    # linear pressure profiles at various time along z line
...
echo "mapping field consistently to \iab{CFD} directory"
...
mapFields ../rasInout -consistent
...
yPlusRAS

if [ $cleancase == "t" ]
  then

    echo "...press enter to clean up case"
    echo "press Ctr+C to keep data"
    read
    ...
```

Code Fragment (C.22) *Post run plotting*

DEM/CFD-DEM Model

Coupled PISO Solver Algorithm

```
Info << "- evolve()" << endl;
particleCloud.evolve(voidfraction,Us,U);

Info << "update Ksl.internalField()" << endl;
Ksl.internalField() = particleCloud.momCoupleM(0).impMomSource();

// particle related fields as well as fluid velocity U are computed at the beginning of the PISO algorithm prior to entering the predictor corrector loop

fvVectorMatrix UEqn
// LHS of NSE
   \alpha\frac{\partial\overline{\mathbf{U}}}{\partial t} + \alpha\nabla\cdot(\overline{\mathbf{U}}\otimes\overline{\mathbf{U}}) + \alpha\nabla\cdot\mathbf{R} - \alpha\nabla\cdot 2\nu\overline{\mathbf{D}} + \mathbf{K}_{sl,\rho}\overline{\mathbf{U}}
(
     fvm::ddt(voidfraction,U)
   + fvm::div(phi, U)
   + particleCloud.divVoidfractionTau(U, voidfraction)
   ==
   - fvm::Sp(Ksl/rho,U)
);
```

```
UEqn.relax();

if (momentumPredictor)
// momentum predictor LHS = RHS
    α ∂U̅/∂t + α∇ · (U̅ ⊗ U̅) + α∇ · R − α∇ · 2νD̅ + K_{sl,ρ} U̅ = −∇p̄ + K_{sl,ρ} U_s
{
    if (modelType=="B")
        solve(UEqn == - fvc::grad(p) + Ksl/rho * Us);
    else
        solve(UEqn == - voidfraction * fvc::grad(p) + Ksl/rho * Us);
}
```

Code Fragment (C.23) *Coupled Navier Stokes Equations*

```
// number of velocity corrections is modified
int nCorrSoph = nCorr + 5 * pow((1 - particleCloud.dataExchangeM().
    timeStepFraction()), 1);

for (int corr = 0; corr < nCorrSoph; corr++)
{
    // rUA   A^{-1}

    volScalarField rUA = 1.0 / UEqn.A();

    // rUAf   A_f^{-1}

    surfaceScalarField rUAf("(1|A(U))", fvc::interpolate(rUA));

    // U   A^{-1}H

    U = rUA * UEqn.H();

    // phi   φ = α(A^{-1}H) · S

    phi = fvc::interpolate(U * voidfraction) & mesh.Sf();

    // phiS   φ_S = αU_s · S

    surfaceScalarField phiS(fvc::interpolate(Us * voidfraction) & mesh.Sf());

    // phiGes   φ_Ges = α(A^{-1}H) · S + A_f^{-1} K_{sl,ρ}(αU_s · S)

    surfaceScalarField phiGes = phi + rUAf * (fvc::interpolate(Ksl/rho) * phiS
    );

    // rUAvoidfraction   αA^{-1}

    volScalarField rUAvoidfraction("(voidfraction2|A(U))",rUA*voidfraction);

    if (modelType=="A")
       rUAvoidfraction = volScalarField("(voidfraction2|A(U))",rUA*
    voidfraction*voidfraction);

    for (int nonOrth = 0; nonOrth <= nNonOrthCorr; nonOrth++)
    {
        // Modified Pressure Correction Equation
          ∇ · (αA^{-1}∇p) = ∇ · [α(A^{-1}H)·S + A_f^{-1} K_{sl,ρ}(αU_s · S)] + ∂α/∂t

        fvScalarMatrix pEqn
```

```
        (
            fvm::laplacian(rUAvoidfraction, p) == fvc::div(phiGes) + fvc::ddt(
    voidfraction)
        );
        ...
        // Modified Flux Corrector Equation
            phiGes = (alpha*A^-1*H)·S + A_f^-1*K_sl,rho(alpha*U_s · S) - (alpha*A^-1*nabla p)

        if (nonOrth == nNonOrthCorr)
        {
            phiGes -= pEqn.flux();
        }
    }
    ...
    // Modified Momentum Corrector Equation
       u = A^-1*H + A^-1*K_sl,rho*U_s - A^-1*nabla p

    if (modelType=="B")
        U -= rUA * fvc::grad(p) - Ksl/rho * Us * rUA;
    else
        U -= voidfraction * rUA * fvc::grad(p) - Ksl/rho * Us * rUA;
        ...
}
```

Code Fragment (C.24) *CFDEM PISO predictor corrector*

LIGGGHTS Modifications

Fix commands specified in the input script require unique sets of argument entries. Input arguments are partially detailed in fragment *C.25*. LIGGGHTS must be rebuilt every time new functionality is introduced. Typically a **.cpp** source code file and its associated **.h** header file are added to the source directory of LIGGGHTS. Each file pair defines a new LIGGGHTS class which is linked into the framework upon subsequent library rebuild. The LIGGGHTS library contains functions which are designed to detect new codes hence no explicit linking specifications are needed and all necessary links are automatically generated when new code is identified.

```
// applied after pair and molecular forces
void FixAcoustic::post_force(int vflag)
{

// function arguments

 A = atof(arg[3]);              // driving amplitude arg[3]
 freq = atof(arg[4]);           // driving frequency arg[4]
 yM = atof(arg[5]);             // particle elastic modulus arg[5]
 cf = atof(arg[6]);             // speed of sound in fluid arg[6]
 rhof = atof(arg[7]);           // fluid density arg[7]

 if (strcmp(arg[8],"x") == 0) { // active acoustic plane normal arg[8]
 plane=0;
 }

 if (strcmp(arg[8],"y") == 0) {
 plane=1;
```

```
 }

 if (strcmp(arg[8],"z") == 0) {
 plane=2;
 }

// region string expected as arg[9]
// region identifier expected as arg[10]

...
// fluid compressibility betaf   β_f = (ρ_f c_f^2)^{-1}

double betaf = 1/(rhof * pow(cf,2.0));

    for (int i = 0; i < nlocal; i++)
    {
    if (mask[i] & groupbit)
        {
        if (iregion >= 0 &&
        !domain->regions[iregion]->match(x[i][0],x[i][1],x[i][2]))
        continue;

    // speed of sound in particle material cp   c_p = √(E ρ_p^{-1})

    double cp = sqrt(yM/rho[i]);

    // particle compressibility beta_p   β_p = (ρ_p c_p^2)^{-1}

    double betap = 1/(rho[i] * pow(cp,2.0));

    // acoustic contrast factor Phi   Φ = [ρ_p + (2/3)(ρ_p − ρ_f)](2ρ_p + ρ_f)^{-1} − (1/3) β_p β_f^{-1}

    double Phi = ((rho[i]+(2/3) * (rho[i] - rhof))/(2 * rho[i] + rhof)) - (
    betap/(3 * betaf));

    // user selects the planar orientation of the acoustic force x, y, z
    // force is added to existing force
      F_x+ = F_{ac,x}
      F_y+ = F_{ac,y}
      F_z+ = F_{ac,z}

    if (plane == 0) {

    // acoustic force on particle in x direction
      F_{ac,x} = A^2 f^3 r_p^3 (2π)^4 Φ ρ_f c_f^{-1} sin(4π f x_p c_f^{-1})

    f[i][0] += pow(A,2.0) * pow(freq,3.0) * pow(r[i],3.0) * pow(2.0 * pi,4.0)
    * Phi * rhof/cf * sin(4.0 * pi * freq * x[i][0]/cf);
            }

    if (plane == 1) {

    // acoustic force on particle in y direction
      F_{ac,y} = A^2 f^3 r_p^3 (2π)^4 Φ ρ_f c_f^{-1} sin(4π f y_p c_f^{-1})

    f[i][1] += pow(A,2.0) * pow(freq,3.0) * pow(r[i],3.0) * pow(2.0 * pi,4.0)
    * Phi * rhof/cf * sin(4.0 * pi * freq * x[i][1]/cf);
            }
    if (plane == 2) {

    // acoustic force on particle in z direction
      F_{ac,z} = +A^2 f^3 r_p^3 (2π)^4 Φ ρ_f c_f^{-1} sin(4π f z_p c_f^{-1})
```

```
    f[i][2] += pow(A,2.0) * pow(freq,3.0) * pow(r[i],3.0) * pow(2.0 * pi,4.0)
    * Phi * rhof/cf * sin(4.0 * pi * freq * x[i][2]/cf);
            }
        }
    }
}
```

Code Fragment (C.25) *Acoustic force*

```
// output format %g-double %d-integer

char *str = (char *) "%g %d %d %g %g %g %g %g %g %g %g %g %g %g";
...

void DumpXYZC::write_frame()
{
  int m = 0;

  for (int i = 0; i < natoms; i++) {

      fprintf
      (
      fp,format,
      simtim[i],                          // simulation time
      iden[i],                            // particle identifier
      types[i],                           // particle type
      radi[i],                            // particle radius
      dens[i],                            // particle density
      coords[m],coords[m+1],coords[m+2],  // particle position (x,y,z)
      vel[m],vel[m+1],vel[m+2],           // particle velocity (vx,vy,vz)
      forc[m],forc[m+1],forc[m+2]         // particle force (fx,fy,fz)
      );

    m += 3;
  }
}
```

Code Fragment (C.26) *Modified dump output format*

```
void CfdDatacouplingMPIcgs::pull(char *name,char *type,void *&from)
{
...
    if(strcmp(type,"vector") == 0) // CFD --> DEM drag and buoyancy forces
    {
    for (int i = 0; i < natomsmax; i++) {
          if ((m = atom->map(i+1)) >= 0) {

            todouble[m][0] = 1e5 * allreduce_long[i][0];
            todouble[m][1] = 1e5 * allreduce_long[i][1];
            todouble[m][2] = 1e5 * allreduce_long[i][2];

            // CFD(N) --> DEM(dynes) conversion factor 1e5
          }
        }
    ...
}
```

```
void CfdDatacouplingMPIcgs::push(char *name,char *type,void *&to)
{
...
    if(strcmp(type,"scalar") == 0) // DEM --> CFD particle radius
    {
    ...
    for (int i = 0; i < nlocal; i++) {
            id = tag[i];

            allreduce_short[id-1][0] = 0.01 * fromdouble[i];

            // DEM(cm) --> CFD(m) conversion factor 0.01
        }
    ...
    else if(strcmp(type,"vector") == 0) // DEM --> CFD particle positions and
    velocities
    {
    ...
    for (int i = 0; i < nlocal; i++) {
            id = tag[i];

            allreduce_long[id-1][0] = 0.01 * fromdouble[i][0];
            allreduce_long[id-1][1] = 0.01 * fromdouble[i][1];
            allreduce_long[id-1][2] = 0.01 * fromdouble[i][2];

            // DEM(cm) --> CFD(m) conversion factor 0.01
            // DEM(cm/s) --> CFD(m/s) conversion factor 0.01
        }
```

Code Fragment (C.27) *CGS $\Longleftrightarrow$ SI conversion*

```
void DumpCustomCGS2SI::pack_mass(int n)
{
...
    buf[n] = 0.001 * rmass[i];
    ...
    buf[n] = 0.001 * mass[type[i]];
    ...
    // CGS(g) --> SI(kg) conversion factor 0.001
}
...
void DumpCustomCGS2SI::pack_x(int n)
{
    ...
    buf[n] = 0.01 * x[i][0];
    ...
    // CGS(cm) --> SI(m) conversion factor 0.01
}

// similarly with y and z

void DumpCustomCGS2SI::pack_vx(int n)
{
    ...
    buf[n] = 0.01 * v[i][0];
    ...
    // CGS(cm/s) --> SI(m/s) conversion factor 0.01
}
```

```
// similarly with vy and vz

void DumpCustomCGS2SI::pack_fx(int n)
{
    ...
    buf[n] = 1e-5 * f[i][1];
    ...
    // CGS(dynes) --> SI(N) conversion factor 1e-5
}

// similarly with fy and fz

void DumpCustomCGS2SI::pack_density(int n)
{
    ...
    buf[n] = 1e-3 * density[i];
    ...
    // CGS(g/cm3) --> SI(kg/m3) conversion factor 1e-3
}
void DumpCustomCGS2SI::pack_radius(int n)
{
    ...
    buf[n] = 0.01 * radius[i];
    ...
    // CGS(cm) --> SI(m) conversion factor 0.01
}
...
```

Code Fragment (C.28) *dump custom/cgs2si*

LIGGGHTS Input Script

A single input script defines the entire discrete system from initialization to run and output. Simulations may be saved and restarted under a different set of conditions read from a separate script. Minor input script additions and omissions are required to accommodate for coupled runs but the general structure remains the same for wet and dry granular systems. The input script **in.acoustic** designed for the study is read by the DEM solver which initializes the case and applies all specified parameters and fixes as necessary. Fixes are in general continually evaluated and updated throughout the simulation whereas initializations are performed once at the beginning of the run. The input script is read line by line scanning in commands as they are encountered, based on expected keywords and associated argument parameters. Certain commands must appear before others. The general template of a LIGGGHTS command line is

```
<command> <name> <group> <keyword1> <arg1.1> <arg1.2> <keyword2> <arg2.1> etc
```

There are a number of **command** words including **fix**, **variable**, **region**, **dump**, etc, each associated with a different function. A user defined **name** is given to each new command and assigned a particle **group**, defaulted to all particles, to which the function is to be applied. All functions have a set of associated **keywords**, some optional and others which have to be included. Each keyword is supplemented by a number of **arguments** which together define the function. Commands exist which do not follow this generalized form. All functions and their expected formats are documented in the

LAMMPS manual [54] and online [61].

There are a number of sections required to define all necessary system parameters. The system initialization section fragment *C.29* is used to define the type of simulation i.e. granular for the LIGGGHTS package, how the domain will be partitioned for parallel computation line *4*, how atom quantities are stored line *2*, which unit system is to implemented line *9*, the type of bounding box boundaries to be used i.e. wrapping, fixed, etc line *6* as well as defining the initial size of the bounding box line *11*, specifying the number of atom types to be used in the simulation, line *12*, specifying particle bin size line *14* and how often particle neighbour lists are to be built and updated line *15*. Bin size has been discussed in section *2.2.2*, and is known to contribute significantly to computation time. Small bins require considerable computing resources.

The material properties section fragment *C.30* is used to define all relevant material and interaction properties. The fragment is well commented and should be referred to if further clarification is needed. Youngs Modulus line *7* is a particle specific material property whereas COF line *18* for example is a contact specific quantity and has to be defined for each contact pair i.e. between particles themselves and between particles and wall. Material properties have been discussed in detail in section *2.2.2*. Reciprocal contact properties i.e. particle to wall and wall to particle, will be considered equal for this study, however this may not necessarily be the case. Since contact phenomena is not of interest in this study the assumption is arbitrary. Material and contact properties are required to compute stiffness and damping coefficients used in force models. The Hertzian history contact model had been selected in this study line *45* with flags 1 and 0 corresponding to friction damp and cohesion flags respectively. The **history** keyword denotes the inclusion of the integral term in the tangential force equation detailed in section *2.2.2*. The friction damp flag 1 corresponds to an included tangential damping term but excluded rolling friction. The 0 cohesion flag corresponds to the exclusion of cohesive forces. If cohesion were to be included then a cohesion energy density proportionality constant would be required to compute this force and must be supplied in the same way as other material and contact properties are. Additional friction damping flag options are selectable which include or exclude various contact terms. Certain flags may require additional material or contact properties as in the case with cohesion i.e. rolling friction requires a rolling friction coefficient. The time step is defined in line *53*.

Fragment *C.31* is used to define wall boundaries and assign material models to them as in line *1*. In this study plane walls were set as in lines *3* and *5* to correspond with the mesh geometry of the developed flow simulation. Periodic boundaries at duct inlet and outlet do not require wall specification. Since particles are inserted into the bulk flow with no cross currents particle wall interaction is not expected, but nevertheless needs to be specified. A number of methods may be used to specify wall boundaries. Simple geometries including cylinders, planes, triclinics and cones as well as combinations thereof may be specified by using in-house commands. More complex geometries may be generated externally using a CAD package and imported in STL format. Imported geometries can be scaled, translated and rotated to position them as required. STL mesh refinement should correspond with particle size i.e. finer STL meshes will interact better with smaller particles. It should also be ensured that any imported STL mesh does not contain triangles with extreme aspect ratios as these could compromise the integrity of particle surface detection algorithms.

Particle insertion is defined in the following section of the input script fragment *C.32*. Particle size and density is defined as uniform or statistically distributed line *3*. Mass insertion fractions may be specified line *5* if a number of previously defined distributions line *3* are to be introduced into a common space. The common space is referred to as an insertion region and must be specified as in line *1*. The region identifier is supplied as an argument to the insertion command line *7*. Insertion may also be assigned to a group of particles, the group being defined prior to the insertion call. In this study a particle group with uniformly distributed size and density ranges was inserted in a region defined ($x \in [0.15 : 0.35], y \in [0.15 : 0.35], z \in [0.1 : 0.11]$) near the flow inlet as illustrated in schematic *3.1*. The 20 particles comprising the discrete system were inserted as stationary i.e. no initial velocity, at the beginning of the coupled run. In general numerous particle insertions are possible, however the common **insert/pack** type had been implemented in the study. Particles are inserted randomly in relation to their seed, selected as 123 for this study. Subsequent simulations with seed 123 had the same particle configuration, which was advantageous in comparing results. The same consideration was made in terms of size and density distributions which also required seeding numbers.

Certain parameters have to be defined which are used as arguments to the acoustic force fix fragment *C.33* as detailed in section *3.3.2*. These include carrier fluid properties density line *10* and speed of sound line *11*, as well as acoustic drive parameters frequency line *5* and amplitude line *6*. Fluid properties were expected to remain constant under the assumption of incompressible, adiabatic and isotropic flow. The function also allows the user to effectively change the carrier fluid through the fluid parameters. Number values may be specified directly as arguments to a fix however creating variable definitions first, as has been done throughout the script, and sending them by reference as fix or other arguments provides clarity and avoids confusion when reviewed. In addition scalar computations may be performed using variable type assignments. Generally the acoustic force acts in a specific region which is defined in line *3*, however in this study the entire simulation domain is activated as a demonstrative means to deflect particles. In design however the transducer length would have to be considered from a cost and implementation point of view. Run time variables which are defined in line *15* were output to screen during a simulation run and updated at interval. Various scalar quantities, pertaining to a single particle identifier 1 in this case, may be output during run time including force, position and velocity which gives the user an indication of simulation integrity. Immediate errors may be identified in this way facilitating timeous termination and correction. Numerous particles may be tracked during run time in this manner, however practical limits exist, hence only a single particle is screen tracked in the current study.

Input script closure is provided in fragment *C.34*. Fixes are implemented in lines *4* to *11* which remain active throughout the simulation run and effectively sustain the particulate system. The acoustic force is specified in line *4* using previously defined variables, all fluid related forces passed to LIGGGHTS including drag and buoyancy are included in the **couple/cfd/force** fix line *5*, gravity is included in line *7*. The coupling interval must be supplied as an argument to the fluid force fix and should correspond with that specified in the **couplingProperties** CFDEM script. The communication scheme is specified **mpicgs**, which implements the modified code applying conversion factors as detailed in section *3.3.2*. The integration scheme is specified in line *9* which serves to update particle positions and velocities. Rayleigh and Hertz time criteria discussed in section *2.2.2* are invoked in line *11* and

computed against the specified time increment.

Screen output is specified in lines *15* to *24*. The **thermo** command line *17* may be used to specify the frequency with which screen output is produced. The **thermo_style** command is used to specify what is to be output as in lines *16* and *23*. The **thermo_modify** command allows for simulation continuation even if atoms are lost, lines *18* and *24*. Screen output may be redefined once particles have been inserted i.e. after the first run step line *21*. The second specification of **thermo_style** is used to invoke particle tracking screen output variables line *23* defined in fragment *C.33*. The system is initialized by running a single time step, thereafter the first dump command is invoked line *30*, which allows initial particle configuration information to be stored. The user defined dump detailed in section *3.3.2* is implemented here which converts the output from CGS units to SI units for consistency with the CFD channel. The dump is invoked using the custom style keyword **custom/cgs2si**. The format is specific in terms of its recognition by Pizza which is used to convert the output into VTK format for viewing inside ParaView. In addition to a specified output format, write interval and dump path are supplied as arguments, write interval being consistent with that specified in the **controlDict** governing the coupled run. The path is required for data exchange purposes, similarly the path to the **in.acoustic** script is supplied in the **couplingProperties** file for this same purpose. The additional user defined dump line *32*, detailed in section *3.3.2* and invoked through keyword **xyzc** is used to produce matrix data defining the particulate system, which may be easily read into MatLab for post processing analysis. Yet another user defined dump line *28* is invoked at a write interval of 1 to record the initial particle configuration. The initial configuration is introduced into the data matrix to form a complete data set for post processing analysis. The remaining data produced at the smaller write interval may be discarded during run time to avoid dealing with large files at a later stage. The CGS unit system is retained in **xyzc** type output and the write interval does not have to correspond to that defined on the CFD side as it is used in a separate analysis stream. Dry granular simulations require an additional run command which is used to define the end time of a simulation, however since intermittent calls to LIGGGHTS are made from OpenFOAM the end time is determined by the specification made in the **controlDict**.

The underlying structure of a DEM input script has been outlined in fragments *C.29* to *C.34*, however various other functions, fixes and settings are accommodated. Simulations may include multiple particle types and interaction characteristics, detailed inter-particle, particle-geometry and external forces depending on the requirement of the simulation, imported, scaled and orientated geometries, thermal effects, as well as bonded particles, atomic lattices and all related forces. Particle insertion coordinates and other particle related properties may be explicitly defined for each, particularly useful when investigating systems containing only a few specific particles.

```
atom_style      granular                                                    1
atom_modify     map array                                                   2
communicate     single vel yes                                              3
processors      1 1 2                                                       4
                                                                            5
boundary        m m p                                                       6
newton          off                                                         7
                                                                            8
```

```
units           cgs

region          reg block 0 50 0 50 0 70 units box
create_box      2 reg

neighbor        1 bin
neigh_modify    delay 0
```

Code Fragment (C.29) *DEM initialization*

```
# defining material and contact properties

# p1 - particle material (quartz)   # wall - wall material (steel)
# yM - Youngs Modulus               # pR - Poissons Ratio
# cF - Coefficient of Friction      # cR - Coefficient of Restitution

variable    yMp1 equal 700e9
variable    yMwall equal 2000e9

variable    pRp1 equal 0.18
variable    pRwall equal 0.3

variable    cRp1p1 equal 0.75
variable    cRp1wall equal 0.5
variable    cRwallp1 equal 0.5
variable    cRwallwall equal 0.7    # n/a

variable    cFp1p1 equal 0.65
variable    cFp1wall equal 0.45
variable    cFwallp1 equal 0.45
variable    cFwallwall equal 0.7    # n/a

# assigning material and contact properties

# template peratomtype 1 2
# 1 particle material property
# 2 wall material property

# template peratomtypepair 2 11 12 21 22
# 11 particle material to particle material contact property
# 12 particle material to wall material contact property
# 21 wall material to particle material contact property
# 22 wall material to wall material contact property

fix     m1 all property/global youngsModulus
    peratomtype ${yMp1} ${yMwall}

fix     m2 all property/global poissonsRatio
    peratomtype ${pRp1} ${pRwall}

fix     m3 all property/global coefficientRestitution
    peratomtypepair 2 ${cRp1p1} ${cRp1wall} ${cRwallp1}
            ${cRwallwall}

fix     m4 all property/global coefficientFriction
    peratomtypepair 2 ${cFp1p1} ${cFp1wall} ${cFwallp1}
            ${cFwallwall}

# contact model assignment
```

```
pair_style      gran/hertz/history 1 0
pair_coeff      * *

# /hertz force model equation set
# /history inclusion of integral term in tangential force
# friction damp flag 1 tangential damping is included, but rolling friction is not
# cohesion flag 0 macroscopic cohesion is excluded

timestep    0.00001
```

Code Fragment (C.30) *Material and contact properties*

```
variable    walltype equal 2

fix    xwalls all wall/gran/hertz/history 1 0 xplane 0 50 ${walltype}

fix    ywalls all wall/gran/hertz/history 1 0 yplane 0 50 ${walltype}
```

Code Fragment (C.31) *Wall boundaries*

```
region    bc block 15 35 15 35 10 11 units box

fix    pts1 all particletemplate/sphere 123 atom_type 1 density
    uniform 2.3 2.7 radius uniform 0.0005 0.0030

fix    pdd1 all particledistribution/discrete 123 1 pts1 1.0

fix    ins1 g1 insert/pack seed 123 distributiontemplate pdd1
    insert_every once overlapcheck yes vel 0 0 0 region bc
    particles_in_region 20 ntry_mc 10000
```

Code Fragment (C.32) *Particle insertion*

```
# acoustic system parameters

region      ac block 0 50 0 50 0 70 units box

variable    freq equal 14794
variable    A equal 0.005

# fluid parameters

variable    rhof equal 1
variable    cf equal 14800

# run time monitoring parameters for particle identifier 1

variable    xposition equal 0.01*x[1]
variable    xforce equal 1e-5*fx[1]
...
```

Code Fragment (C.33) *Variable definition*

```
# fix specification

fix   ac1 all acoustic ${A} ${freq} ${yMp1} ${cf} ${rhof} x region ac

fix   cfd1 all couple/cfd/force every 1 mpicgs

fix   gravi all gravity 981 vector 0.0 0.0 -1.0

fix   integr all nve/sphere

fix   ts all check/timestep/gran 5000 0.2 0.2

# screen data output specification

compute             1 all erotate/sphere
thermo_style        custom step atoms
thermo              5000
thermo_modify       lost ignore norm no
compute_modify      thermo_temp dynamic yes

run     1

thermo_style        custom step atoms v_xposition v_xforce v_zposition
              v_xforce v_yforce v_zforce
thermo_modify       lost warn norm no

# log specification

dump    dumpInit all xyzc 1 ../DEM/post/xyzcDataInit

dump     dmp all custom/cgs2si 5000 ../DEM/post/dump.acoustic id type
    type x y z ix iy iz vx vy vz fx fy fz omegax omegay omegaz          radius

dump    vardump all xyzc 500 ../DEM/post/xyzcData
```

Code Fragment (C.34) *Run specification*

Files for Coupled Simulation

'0' Directory

The coupling code requires additional discrete fields including particle velocity **Us**, momentum exchange **Ksl**, fluid particle volume fraction **voidfraction** and fluid density **rho**. The relevant equations containing the field variables have been outlined in sections *2.2.3* and *3.3.1*. The coupling code is designed for incompressible RANS or LES type simulations. All mapped and added fields are included in the **0** directory for the coupled simulation. Coupled simulations must be run from time zero hence mapped fields need to be carried through to a separate **0** folder designated for coupling simulation.

All added field boundaries are initialized with **zeroGradient** entries as in fragment *C.35* lines *5* to *15*. Internal fields are specified in accordance with physical quantities. The **Ksl** field for discrete momentum exchange at a cellular level is zero initialized as no particles are present at the onset of simulation fragment *C.35*. fluid density **rho**, is constant under the assumption of incompressibility,

stated at the beginning of this document and used to formulate all relevant governing equations, fragment *C.36*. Discrete particulate phase velocity **Us** at a cellular level is initialized at zero as no particles are present at the onset of simulation fragment *C.37*. Dimensionless discrete normalized fluid particle volume fraction **voidfraction** at a cellular level is initialized at unity as there are no particles present at the onset of simulation fragment *C.38*.

```
dimensions      [1 -3 -1 0 0 0 0];

internalField   uniform 0;
...
    INLET
{
        type            zeroGradient;
}
    OUTLET
{
        type            zeroGradient;
}
    WALL
{
        type            zeroGradient;
}
```

Code Fragment (C.35) *Ksl*

```
dimensions      [1 -3 0 0 0 0 0];

internalField   uniform 1000;
```

Code Fragment (C.36) *rho*

```
dimensions      [0 1 -1 0 0 0 0];

internalField   uniform (0 0 0);
```

Code Fragment (C.37) *Us*

```
dimensions      [0 0 0 0 0 0 0];

internalField   uniform 1;
```

Code Fragment (C.38) *voidfraction*

'constant' Directory

A **couplingProperties** script, fragment *C.39*, located in the **constant** directory, must also be completed prior to running coupled simulations. The script is used to define all settings to be implemented in the coupling procedure. The coupling model is selected as either **A** or **B**. Model **A** is more sensitive to void fraction as mentioned in previous discussions. The system in question was dilute hence void fraction was expected to remain at constant unity with negligible effect on the governing equations, rendering the choice of model arbitrary. Coupling interval was based on relative CFD and DEM time increments, which in this study was kept at unity. This 1:1 coupling interval was possible because the DEM time increment was relatively large. Generally the CFD timestep is lowered to meet that required of the DEM code to adequately resolve particulate collision behaviour. In most cases the CFD time increment cannot be reduced to levels required by the DEM code as it becomes computationally implausible. This results in a mismatched timestep and a coupling interval greater than unity. DEM time step is always expected to be smaller than or equal to that used to simulate CFD. This large DEM time increment was essentially inadequate to resolve particle collision behaviour however dilute system collision behaviour does not affect the flow in any way and is relatively unimportant in studying the effects of the acoustic force on particle deflection. Reducing both CFD and DEM time increments would considerably increase computation time. Coupling intervals reported in literature range between 10 and 100 [20, 25]. Depending on the system at hand either the CFD or the DEM timestep may consume the bulk of simulation time in which case relative time increments may be revised to optimize the simulation in terms of computation time. Larger coupling intervals may result in inaccurate fluid particle exchanges.

The locate model is used to specify the method of identifying mesh cells containing particles. The **centre** locate model considers cell centres as definitive. The meaning of the data exchange model is unambiguous. Other models exist including **oneWayVTK** and **twoWayFiles** however the most common method is the **twoWayMPI** implemented in this study. The method selected here must be consistent with that implemented in the LIGGGHTS input script. Associated model coefficients are provided below model definitions, lines *30* to *49*. The properties are often used to specify file names of field variables used to compute for example particle forces, and also file paths to specify where particle output data is located. A variety of force models detailed in section *2.2.2* can be selected from, as in lines *13* to *21*, however only the drag and buoyancy forces had been accounted for in this study. An additional gravity field file **g**, required by the **Archimedes** buoyancy model is included in the **constant** directory containing a single gravity vector specification of magnitude, direction and dimension.

A **liggghtsCommands** script is included in the **constant** directory, used to produce the LIGGGHTS run command at the specified coupling interval. Two models are selectable, which are used to determine stop start behaviour of the DEM simulation. The settings defined in the script relate to streamlining communication paths between successive calls to LIGGGHTS, and would have to be correctly specified to account for any changing conditions in the DEM script. These changing conditions would require complete script revision by the solver, whereas it would otherwise perform only partial run-throughs post initialization. The DEM script used in this study required no special treatment and a standard

method used in tutorials had been adopted.

```
modelType "B";

couplingInterval 1;

voidFractionModel centre;

locateModel standard;
...
dataExchangeModel twoWayMPI;

averagingModel dilute;

forceModels
(
    DiFeliceDrag
    Archimedes
    //SchillerNaumannDrag
    //MeiLift
    //virtualMassForce
    //gradP
);

momCoupleModels
(
    implicitCouple
);

turbulenceModelType "RASProperties";
...
implicitCoupleProps
{
    velFieldName "U";
    granVelFieldName "Us";
    voidfractionFieldName "voidfraction";
    KslLimit 50000;
}
DiFeliceDragProps
{
    velFieldName "U";
    densityFieldName "rho";
}
...
ArchimedesProps
{
    densityFieldName "rho";
    gravityFieldName "g";
}
...
twoWayMPIProps
{
    maxNumberOfParticles 10000;
    liggghtsPath "../DEM/in.acoustic";
}
```

Code Fragment (C.39) *couplingProperties*

'system' Directory

All CFDEM coupling codes are run in parallel hence the need for domain partition. The **decomposeParDict** script fragment *C.40* serves to define the partition of simulation domain amongst the available processors line *1*. Processor partitions may be redefined in the DEM input script however this is not mandatory as the decomposition is assumed from that specified in the **decomposeParDict** script. The domain should be distributed so that each processor deals with similar numbers of particles. This consideration is however more pertinent when simulating systems with many particles. For dilute systems, as in this study, decomposition is arbitrary, nevertheless it must be specified. A number of domain decomposition methods exist for more complex geometries however the **simple** method was chosen in this study because of the regular geometry. The **manual** decomposition type for example allows separate data file allocation of individual cells to processors. The domain was decomposed into two segments along the z dimension as in line *7*. The cell skew factor line *8* is retained at the recommended default value. Any decomposition type selected as in line *3* is accompanied by a set of coefficients or supplementary entries which parametrize that method.

Entries in the **controlDict** script were adjusted to run a single uninterrupted simulation from time zero at a greatly reduced timestep fragment *C.41*. CFD and DEM time increments were set equal to one another. The end time is adjusted according to flow velocity i.e. the time required for the particles inserted at the inlet of the control volume to reach the outlet. This produced equivalent particle travel lengths under all simulated velocities facilitating comparative post processing analysis. Inlet and outlet boundaries had been specified as periodic in the DEM script and would, in the absence of time limiting result in feeding back of particles at the inlet. Write interval was also set equal to that defined in the DEM run script so that snapshots were generated at corresponding times for post processing purposes.

Additional divergence and Laplacian schemes had to be defined as no defaults had been provided fragment *C.42*. These accounted for additional terms introduced through the coupling procedure which had to be computed. Standard schemes had been selected for these missing operations.


```
numberOfSubdomains 2;                                                    1
                                                                         2
method          simple;                                                  3
                                                                         4
simpleCoeffs                                                             5
{                                                                        6
    n               ( 1 1 2 );      // domain decomposition (x,y,z)       7
    delta           0.001;          // cell skew factor                  8
}                                                                        9
```

Code Fragment (C.40) *decomposeParDict*

```
...                                                                      1
startFrom       startTime;                                               2
                                                                         3
startTime       0;                                                       4
```

```
stopAt          endTime;

endTime         1.0;

deltaT          0.00001;
...
writeInterval   5000;
...
```

Code Fragment (C.41) *CFDEM controlDict*

```
...
divSchemes
{
    ...
    div((viscousTerm*dev(grad(U).T()))) Gauss linear;
}

laplacianSchemes
{
    ...
    laplacian(viscousTerm,U)            Gauss linear corrected;
    laplacian((voidfraction2|A(U)),p)   Gauss linear corrected;
}
```

Code Fragment (C.42) *CFDEM fvSchemes*

cfdemRun Script

The **cfdemRun.sh** script was used to call the coupling solver **cfdemSolverPiso_shared** and initiate the coupled simulation. The required solver arguments are defined prior to the call and referenced into an argument list fragment *C.43* line *8*. The solver decomposes the simulation domain, runs the solvers intermittently in parallel and finally reconstructs the case by amalgamating processor results. Separate simulation folders are produced for each processor.

CFD and DEM output files have to be converted to VTK format for visualization in ParaView. The corresponding operations are found in lines *12* and *16*. DEM data conversion is done through a set of Pizza commands which read the dump file produced in the **post** folder, scale the results and write VTK files for particle positions and their containing bounding box. CFD data conversion is done through the OpenFOAM post processing utility **foamToVTK** which formats field variable data for every time folder written during the simulation run. The coupled run may be visualized by running paraview. The **-dr** option is included to circumvent a version bug by removing any defaults which may have been set in a previous session. The user is then prompted to clean the case or retain the data.

```
...
pizzaPath="$CFDEM_PIZZA_DIR"
```

```
casePath="$(dirname "$(readlink -f ${BASH_SOURCE[0]})")"
solverName="cfdemSolverPiso_shared"
nrProcs="2"
...

parCFDDEMrun $logpath $logfileName $casePath $headerText $solverName $nrProcs
    $machineFileName $debugMode
...

cd $casePath/DEM
python -i $pizzaPath/pizza.py -f AcousticPizza
...

cd $casePath/CFD
foamToVTK

paraview -dr

# clean case
echo "...press enter to clean up case"
echo "press Ctr+C to keep data"
read
...
```

Code Fragment (C.43) *cfdemRun.sh*